

195 PTAS.
(IVA Incluido)
mi computer
105
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
EPSON SQ-2000

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen IX-Fascículo 105

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones
Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Aribau, 185, 1.°, 08021 Barcelona
Tel. (93) 209 80 22 - Télex: 93392 EPPA

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 120 fascículos de aparición semanal, encuadernables en diez volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-181-X (tomo 9)
84-85822-82-X (obra completa)
Depósito Legal: B. 52-84

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 218601
Impreso en España-Printed in Spain-Enero 1986

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (120 fascículos más las tapas, guardas y transferibles para la confección de los 10 volúmenes) son las siguientes:

a) Un pago único anticipado de 27 105 ptas. o bien 10 pagos trimestrales anticipados y consecutivos de 2 711 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S. A. (Aribau, 185, 1.°, 08021 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

No se efectúan envíos contra reembolso.





En la pista

En esta nueva serie dedicada a los micros y los juegos de apuestas analizaremos, en primer lugar, cuatro paquetes de carácter hípico muy populares en Gran Bretaña



Ayudar a la suerte Distintos factores inciden en el resultado de una carrera favoreciendo un enfoque anárquico y no rentable a las apuestas. El ordenador actúa como una útil herramienta para el apostador serio, reduce la inversión y maximiza la ganancia potencial al detectar combinaciones acertadas y posibilidades atrayentes

Sporting Pictures Ltd.

Si pensamos en la inmensa popularidad de los juegos de apuestas, casi no sorprende saber que la introducción de los ordenadores haya llevado a muchos programadores británicos a idear algoritmos destinados a maximizar las ganancias, minimizar las pérdidas y predecir los resultados de las distintas carreras de caballos que se realizan en Gran Bretaña. En esta serie de cuatro capítulos examinaremos algunas de las técnicas y la teoría en las que se basan las aplicaciones de apuestas por ordenador, y adelantaremos nuestra opinión acerca de las posibilidades de que un micro pueda convertir a su usuario en millonario.

En este capítulo examinamos cuatro paquetes distintos, de gran aceptación en Gran Bretaña, cada uno de los cuales afirma ser de utilidad para el apostador. Asimismo damos una mirada a una de las principales fuentes de información para el apostador interesado en utilizar tales programas, la publicación *The Sporting Life* (que se podría traducir como «Vida de apuestas») y evaluamos los márgenes probables de beneficios (¡o pérdidas!) para cada título.

«Coursewinner»

Uno de los programas más interesantes que tienen a su disposición actualmente los apostadores británicos es el *Coursewinner* (Ganador de carreras), de Selec Software. En él se da por sentado que se dispone del *The Sporting Life* o de un periódico similar especializado en carreras. Los factores escogidos para el análisis son razonables y es probable que den una evaluación exacta.

Una característica muy grata es la capacidad de entrar hasta tres clasificaciones de velocidad tal como se publican en la prensa popular o deportiva o se obtienen de especialistas en clasificaciones como el «Timeform».

El programa, activado por menú, es fácil de usar aunque posee una detección de errores bastante limitada. Un detalle bien pensado es la provisión de un arranque en caliente sin pérdida de datos actuales, de modo que pulsar ESCAPE (en la versión BBC) de forma accidental no nos supone ningún desastre.

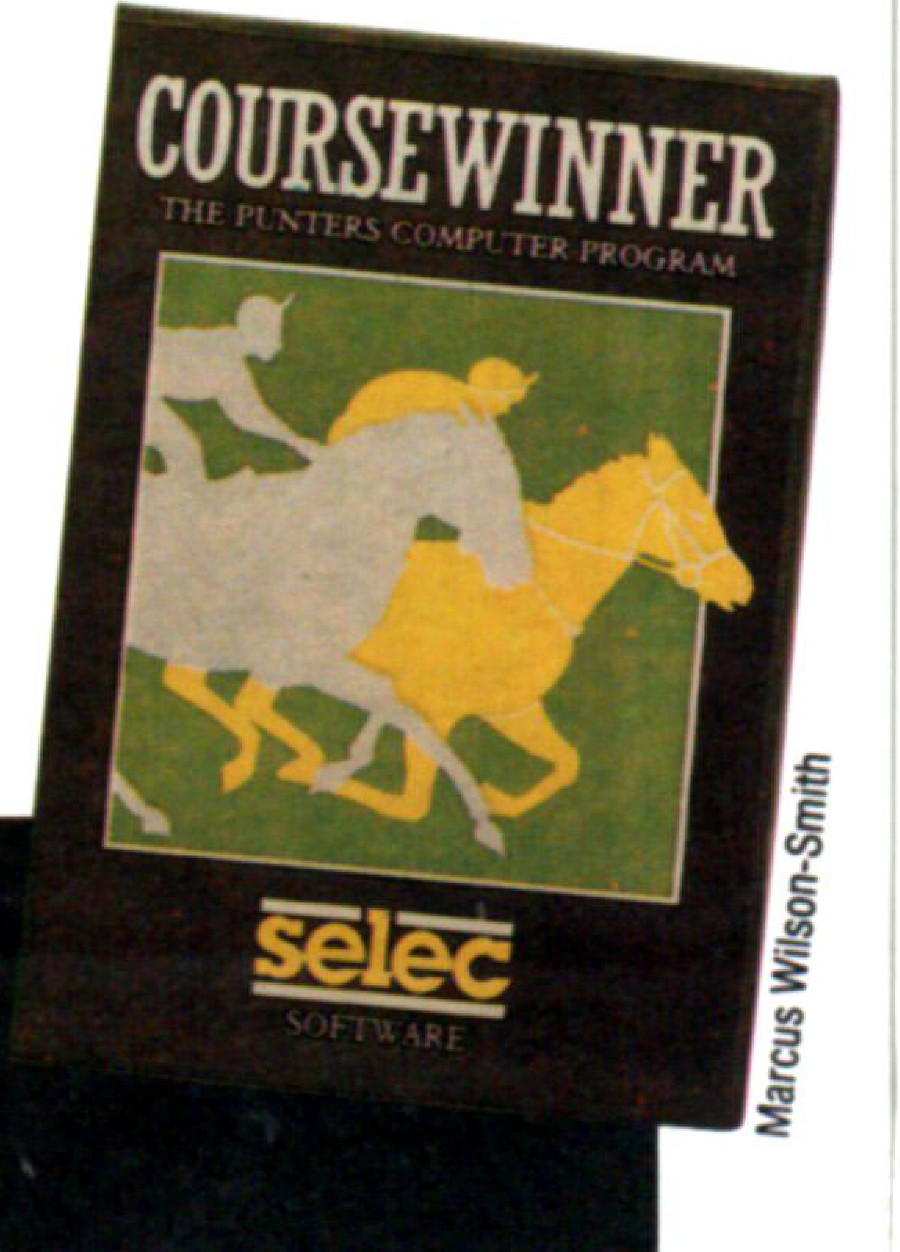
Una vez entrados los factores de forma correspondientes, se espera que el usuario proporcione las cotizaciones reales en oferta, porque el programa, además de aventurar las verdaderas posibilidades del caballo, intenta aconsejar si las de los corredores de apuestas representan un buen valor, en cuyo caso recomienda una apuesta «tres estrellas». Sin embargo, dado que el mercado de apuestas sólo se forma unos diez minutos antes del inicio de cada carrera, esto no siempre es posible. En cambio, puede entrarse el pronóstico de apuestas de los diarios matutinos, invalidando en consecuencia las apuestas «tres estrellas» recomendadas, pero no necesariamente las clasificaciones.

Otra opción implica el ajuste de la estimación dada para los diversos factores de forma al calcular las posibilidades de un caballo. Ésta también es una configuración útil, en particular para el apostador experimentado.

El paquete se probó con el caballo *Glorious Goodwood* en 18 carreras de diez corredores o menos. Las apuestas de valores «tres estrellas» no fueron espectaculares, con tres ganadores en 12 selecciones con una ganancia de premios pareja, descontados los impuestos, de un premio de entre £1,60 y £1.

El *Coursewinner* es un programa interesante y bien diseñado, basado en principios sólidos, que sobre la base de una muestra estadística de reconocida solvencia, parece ofrecer al apostador cauteloso la posibilidad de apostar de forma selectiva y rentable.

Obviamente, el mejor consejo para quien intente utilizar seriamente este sistema, u otro, es que lo pruebe concienzudamente antes de invertir dinero auténtico.



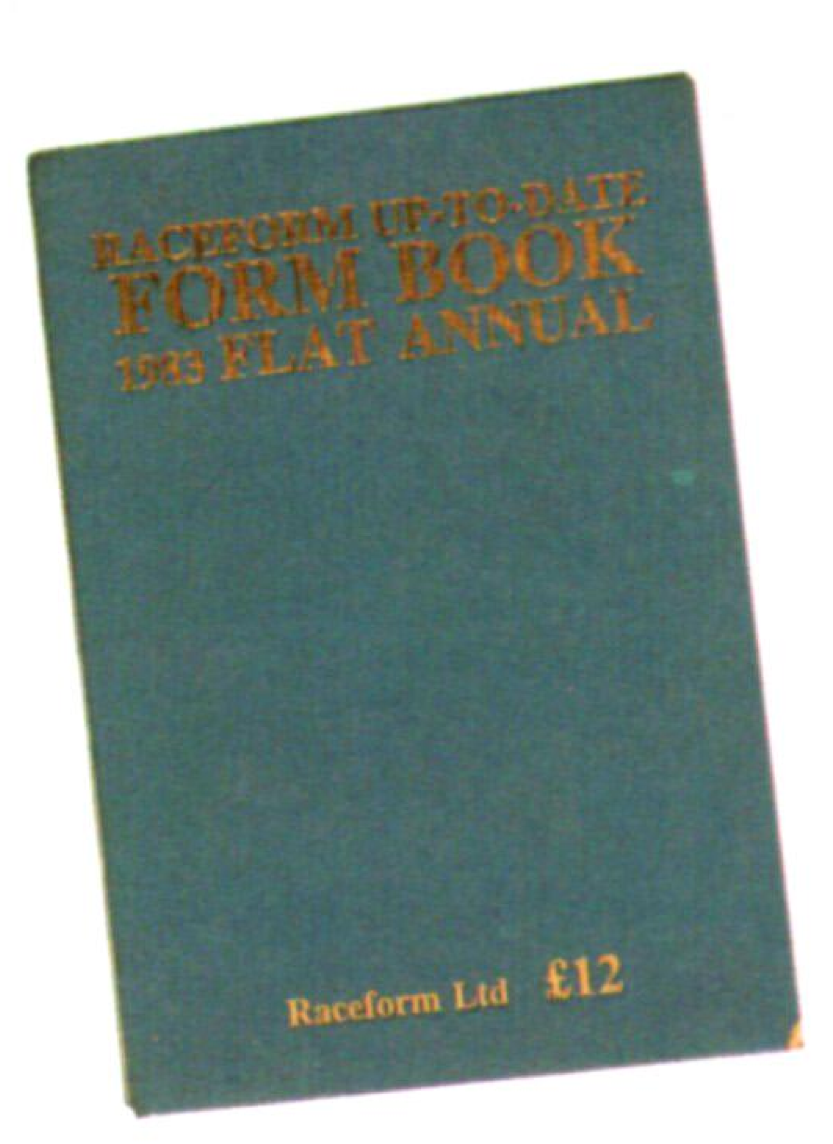
Marcus Wilson-Smith

Coursewinner: Para el Spectrum, Commodore 64, gama Amstrad CPC, Atari, Apple II y BBC Modelo B Micro
Distribuido por: Selec Software, 37 Councillor Lane, Cheadle, Cheshire, Gran Bretaña

Usando el "Coursewinner"

Al cargar el *Coursewinner*, se presenta un menú; digitando C aparece una lista de pistas. Se digita el número de la pista en la cual se correrá la carrera a analizar y después se entra la distancia para el hipódromo más cercano y el *going* oficial (que está en la parte superior de la Racecard). El programa retorna al menú principal. Se digita I para entrar los datos de la carrera. El programa se bifurca a un submenú con cuatro rutinas para entrar los diferentes tipos de datos requeridos. Hay que pasar por cada rutina entrando los datos en la forma en que lo solicita el programa. En la sección titulada "factores de velocidad" se deben utilizar tres de estos sistemas de clasificación publicados: Stopwatch (*Sporting Life*), Form Ratings (*Sporting Life*), Spotform (*Daily Mirror*), Formcast (*Daily Mail*) y Timeform (por suscripción). De ellos, se recomiendan las clasificaciones del "Timeform". Al retornar al menú principal, hay que digitar P. Aparecerá una solicitud de cotizaciones de partida. Si no dispone de un informe de las agencias de apuestas, use el pronóstico de apuestas; pero recuerde que devaluará las apuestas "tres estrellas" recomendadas. No invalidará el cálculo hecho por el programa de las "auténticas posibilidades" de los corredores.

Cuando reaparezca el menú principal hay que digitar un análisis de la carrera. Cuanto más escasas sean las posibilidades calculadas, tanto mejores serán las probabilidades del caballo en cuestión. Es probable que valga la pena restringir las apuestas a caballos con posibilidades calculadas de 1-1 (dinero saldado) o mejores, dado que éstos ganarán el 50 % de sus carreras con algunas carreras largas como perdedor. Si los corredores de apuestas ofrecen por ellos posibilidades de 6-4 o más, tanto mejor



«Sprint formula»

Brimardon Computer Racing Service ofrece numerosos servicios al apostador británico cuidadoso que desee aprovechar mejor su micro. Éstos incluyen guías para que el usuario escriba sus propios programas, programas ya escritos que se pueden adaptar a las necesidades individuales, programas diseñados a la medida de las especificaciones del cliente y el *Sprint formula* (Fórmula de los *sprints*). Este último es un paquete dirigido al apostador experimentado que conoce el valor en *sprints* de las cifras de velocidad basadas en tiempos de carrera. Se centra en los handicaps de todas las edades hasta siete estadios (un estadio=201) porque estas carreras de notoria dificultad ofrecen apuestas abiertas. Los autores arguyen, con cierta justificación, que esto proporciona una oportunidad de oro para el apostador que posee un medio fiable de evaluar los méritos.

La cifra de velocidad del caballo, que es vital para la fórmula, se puede mejorar o devaluar según el *going* (estado de la pista), la distancia y la clase de carrera en la que se obtuvo la cifra. Además, se juzgan las cifras de forma reciente y se da una estimación en relación a indicadores tan obvios como un segundo lugar obtenido la última vez. Los autores consideran que estos corredores a menudo parten con posibilidades restringidas que no ofrecen ninguna expectativa. El objeto es tomar ganadores con mayores probabilidades.

El sistema, por cierto, hace esto para quienes estén preparados para apoyar a los dos caballos mejor clasificados. Brimardon ofrece detalles de los resultados de la temporada, que se pueden comprobar. Hasta ahora, los dos mejores han producido 28 ganadores en 123 apuestas para una ganancia de premios pareja, descontados impuestos, de 79 puntos, lo que representa un 62 % sobre el movimiento total.

Los autores admiten que, debido a limitaciones de memoria, el resultado del *Sprint formula* no es muy bueno, mientras que las rutinas de entrada son un poco chapuceras. La detección de errores es inexistente y queda en las manos del usuario evitar la entrada de datos erróneos. En compensación, el programa viene con una documentación extensa y eficaz. En conjunto, es un paquete útil para el apostador selectivo, que valorará su inteligencia y su, al parecer, feliz aproximación al problema de la verdadera rentabilidad de las apuestas.

Sprint formula: Para el BBC Modelo B, Acorn Electron, Spectrum, Commodore 64, gama Amstrad CPC, Tandy, Dragon, Oric y Atari
Distribuido por: Brimardon Computer Racing Service, 48 Pierremont Road, Darlington, Gran Bretaña



"Z5 horse race forecast"

El *Z5 horse race forecast* (Pronóstico de carreras de caballos Z5), del profesor Frank George, toma sus datos de *The Sporting Life* y utiliza cifras de velocidad del *Raceform Handicap Book*. Emplea menos datos que el *Coursewinner* 3 y acepta dos clasificaciones: la cifra de velocidad adecuada más el servicio de clasificaciones de forma de *The Sporting Life*.

La información se entra en respuesta a una serie de avisos. Se detectan los errores más obvios y, antes de entrar cada dato, se interroga acerca de su veracidad. El programa clasifica entonces a cada caballo de acuerdo a la siguiente escala: apuesta excelente, muy buena, posible, pobre, de caballo no favorito y de eliminado. No se realiza ningún intento para cuantificar estas clasificaciones, de modo que es obligación del usuario probar metódicamente el programa con el fin de establecer su validez estadística.

La desconcertante característica de este programa es la frecuencia con que se da, hasta cinco corredores de la misma carrera, las clasificaciones máximas en conjuntos de diez corredores o menos. El autor hace dos recomendaciones: o no apostar, o apoyar a todos los caballos de mejor clasificación (si es posible hacerlo conservando, así y todo, alguna ganancia). Sin embargo, el número de verdaderas posibilidades de ganar sobre esta base es limitado.

Z5 horse race forecast: Para el Spectrum, BBC Micro Modelo B y Commodore 64
Distribuido por: Bureau of Information Science, Chalfont, St. Giles, Gran Bretaña

"Hulk" 1 y 2

El *Hulk* (de Warm Boot Ltd) no está diseñado específicamente como ayuda para el apostador por ordenador. Su creador lo describe como un "ingeniero con el conocimiento de un ignorante", en el sentido de que ayuda al usuario a construirse su propio sistema experto. Si se acepta que el apostador con éxito es un experto en la materia, ¿por qué no tratar de imbuirle esa experiencia al ordenador? En consecuencia, *Hulk* es útil para el apostador experimentado a quien le guste trabajar con "sistema": unas reglas basadas en el análisis de datos pasados que, aplicados con rigor, crea ganadores en número suficiente para obtener beneficios.

La primera tarea, y la que más tiempo lleva, es la de preparar una base de datos. En la versión para el BBC Micro, las limitaciones de memoria requieren ingenio por parte del usuario, quien debe tener una idea anticipada y clara de los factores que con toda probabilidad serán relevantes, y seleccionar luego un tipo de carrera específica para el análisis. Una estrategia típica podría suponer el estudio de handicaps para una distancia dada en carreras de diez corredores o menos, concentrándose en los primeros tres o cuatro del pronóstico de apuestas. Es esencial asegurarse escoger una muestra representativa desde el punto de vista estadístico.

Los datos son analizados por un programa llamado *Look*, que permite desarrollar un conjunto de reglas (típicamente siete u ocho) que predicen una hipótesis dada (en este caso, si un caballo es o no ganador). Tras haber obtenido un conjunto de reglas satisfactorio, las mismas se pueden aplicar a acontecimientos futuros mediante otro programa, denominado *Leap*, que realiza sus predicciones desde el punto de vista del porcentaje de probabilidades.

El *Hulk* es un programa fascinante para el apostador "sistemático" dedicado al análisis estadístico como medio para hallar ganadores. Utilizándolo, es posible idear conjuntos de reglas que pronostiquen ganadores sobre una base muy selectiva, pero con un grado de exactitud que supera el 60 %.



Hulk I y II: Hulk I paara el BBC Modelo B Micro; Hulk II para máquinas dBase II, MS-DOS y PC-DOS
Distribuido por: Warm Boot Ltd, Finsbury Business Centre, 40 Bowling Green Lane, London EC1R ONE, y por Brainstorm Computer Solutions, 103a Seven Sisters Road, London, N7 7QN, Gran Bretaña

Para informarse

La Racecard da la hora, la distancia, las condiciones y el valor de la carrera seguidos por una lista de corredores. Leyendo de izquierda a derecha, los detalles para cada corredor son los siguientes: número, posición en las seis últimas carreras, nombre, propietario, entrenador, edad, peso con carga, jockey y *draw*. De vez en cuando también se encontrará información complementaria sobre penalizaciones, ganadores de pistas y distancia, y ventajas para aprendices. A la lista de corredores le siguen las "clasificaciones de cronómetro". El mejor tiempo registrado por un caballo se expresa en forma numérica simple tras haber efectuado varios cálculos teniendo en cuenta *going*, peso, etc. Es importante una buena cifra de velocidad, ya que no tiene sentido apoyar caballos lentos. El pronóstico de apuestas son las ideas del experto acerca de cómo es probable que apuesten los apostadores. Es una guía orientativa y no representa lo que ocurra en realidad cuando se forme un mercado.

El Sporting Life Form de la derecha da el perfil general del caballo, incluyendo sus posiciones durante la temporada, nombre, edad, peso con carga, color, sexo, crianza, lista de carreras ganadas (incluyendo distancia, *going* y pista), triunfos totales y triunfos de la temporada. A continuación siguen unas breves notas que describen las tres últimas carreras

The Racecard

Fourth Race



One mile and a half, for three yrs old and upwards
4.10 THE ALYCIDON STAKES
(Listed Race)

£12000 added to stakes
Distributed in accordance with Rule 194 (iii)(a)
(Includes a fourth prize)
for three yrs old and upwards which, at starting, have not won a Pattern race since two yrs old
ONE MILE AND A HALF
£24 to enter, £96 extra if declared to run

Weights: 3-y-o colts and geldings....8st 2lb; fillies 7st 13lb
4-y-o and up colts and geldings.....9st 1lb; fillies 8st 12lb
Penalties, since 2-y-o, a winner of a race value £4000 .... 3lb
Of a race value £8000 .... 6lb
Allowances: 4-y-o and up which, at starting, have not won a race since 1983 and 3-y-o maidens at starting 6lb
' A trophy value £500, at the option of the winner, is included in the value of this race

A SS

48 entries, 31 at £24 and 17 at £120.-Closed July 17th, 1985

Owners Prize Money. Winner £7297; Second £2393; Third £1136 ; Fourth £508
(Penalty Value £9489.60)

Form			Trainer	Age	st	lb	Draw
11—2231 D	2	**PARLIAMENT** .... Ch h Lord Gayle (USA) - Harbrook. **Mrs Pamela Stokes** .... (C. R. Nelson, Upper Lambourn). ROYAL BLUE and WHITE stripes, RED sleeves.		5	9	7	(4) P. Cook
211245—	4	**SHERNAZAR** .... B c Busted - Sharmeen (FR). **H.H. Aga Khan** .... (M. R. Stoute, Newmarket). GREEN, RED epaulets.		4	9	1	(2)

(Breeder - Shanbally House Stud.)
Very consistent sort. Victory in listed race last time (Ayr 10f) and close up 2nd in Group 3 Westbury Stakes, (Sandown 10f) behind Elegant Air illustrates that he is not without claims.

(SHERNAZAR) **(Breeder - H.H. Aga Khan.)**
Had some very smart form last year, chasing home Commanche Run in the Gordon Stakes here 12f and

SPORTING LIFE FORM

KEY: a—slower than average; b—faster than average; eq—equals average; bl—blinkers; d—disqualified; ow—overweight; £—value to the winner; asterisk denotes apprentice claim, figures appearing after the jockey's name before bracket denote draw position.

2.30—NEWHOLME STAKES (2-y-o). 6f. (£1,244).

AVENTINO (8-11) ch c Cure The Blues — Sovereign Dona by Sovereign Path.

00 BROKERS DREAM (April 7, 5,400gns) (8-11) ch c Free State — Pamora by Blast.

April 27, Sandown, 5f (2-y-o) mdn, good, £2,691: 1 Teetoy(USA) (9-0, 8); 2 Cliveden(USA) (9-0, 11); 3 Sit This One Out (9-0, 10); 10 BROKERS DREAM (9-0, B Thomson, 7), in touch to half-way (33 to 1). 12 Ran. 3l, 1½l, hd, ¾l, 1l. 1m 3.19s (a 2.69s).

April 16, Newmarket, 5f (2-y-o), good, £2,532: 1 Andartis (9-0 , 11); 2 Meadow Moor (9-0 , 8); 3 For Dear Life (9-0 , 5); 9 BROKERS DREAM (9-0 , B Thomson , 7), never beyond mid-division (14 to 1 tchd 20 to 1). 11 Ran. 1½l, nk, ¾l, nk, 2½l. 1m 2.23s (a 2.43s).

3 CONQUERING HERO(USA) (May 2) (8-11) b c Storm Bird — Wave In Glory by Hoist The Flag.

Aug 3, Windsor. See ELNAWAAGI(USA).

311 ELNAWAAGI(USA) (Feb 4, $ 4,000,000) (9-6) b c Roberto(USA) — Gurkhas Band (USA) by Lurullah. **1985, 6f good (Windsor), 6f good to firm (Thirsk). £2,612.**

Aug 3, Windsor, 6f (2-y-o), good, £941: 1 ELNAWAAGI(USA) (9-4 , A Murray , 12), made virtually all, stayed on well when shaken up final furlong (5 to 4 fav op 5 to

41 FEISTY (March 13, 44,000gns) (9-3) b c Pitskelly — Golden Empress by Cavo Doro. **1985, 6f good (Yarmouth). £1,066.**

Aug 8, Yarmouth, 6f (2-y-o) mdn, good, £1,066: 1 FEISTY (9-0 , T Ives , 1), made virtually all, drew clear well over one out, comfortably (100 to 30 op 5 to 2 tchd 7 to 2); 2 Below Zero (9-0 , 5); 3 Pulham Mills (9-0 , 6). 7 Ran. 6l, ½l, ½l, 7l, ½l, 1½l. 1m 16.8s (a 4.2s).

June 17, Windsor, 5f (2-y-o), good to firm, £965: 1 Roaring Riva (8-11 , 14); 2 Dee-Jay-Ess (9-3 , 1); 3 Sitzcarraldo (8-4 , 7*, bl, 8); 4 FEISTY (8-11 , T Ives , 9), always chasing leaders, stayed on inside last (11 to 1 op 5 to 1 tchd 12 to 1); 0 PORTHMINSTER (8-11 , J Reid , bl, 11), (50 to 1). 17 Ran. Sht hd, 2l, 2½l, 2½l, ¾l. 59.9s (a 0.6s).

00 FIRST BILL (Feb 28) (8-11) ch c Nicholas Bill — Angelica by Hornbeam.

Aug 2, Goodwood, 7f (2-y-o) mdn, good, £4,675: 1 New Trojan (9-0 , 11); 2 Mashkour(USA) (9-0 , 8); 3 Bronze Opal(USA) (9-0 , 13); 8 FIRST BILL (9-0 , J Matthias , 9), well placed for over 4f, weakened (50 to 1). 15 Ran. ¾l, 3l, 3l, 1½l, ½l. 1m 31.59s (a 4.99s).

July 13, Salisbury, 7f (2-y-o) mdn, good to firm, £1,697: 1 Atig(FR) (9-0, 17); 2 Ininsky (9-0, 1); 3 Sohail(USA) (9-0, 3); 0 FIRST BILL (8-7 , G Landau , 7*, 12), speed for 4f (33 to 1 op 10 to 1). 19 Ran.

Cortesía de The Sporting Life

Potenciar los personajes

Centremos nuestra atención en las rutinas de manipulación de objetos que ya hemos analizado

Las líneas que añadiremos a nuestro módulo básico en el último capítulo permiten que los personajes recojan objetos y, en grado limitado, los utilicen. Antes de seguir adelante diseñando los restantes árboles de decisión para controlar la interacción de los personajes y de la trama, es interesante examinar con mayor detalle el proceso de manipulación de objetos.

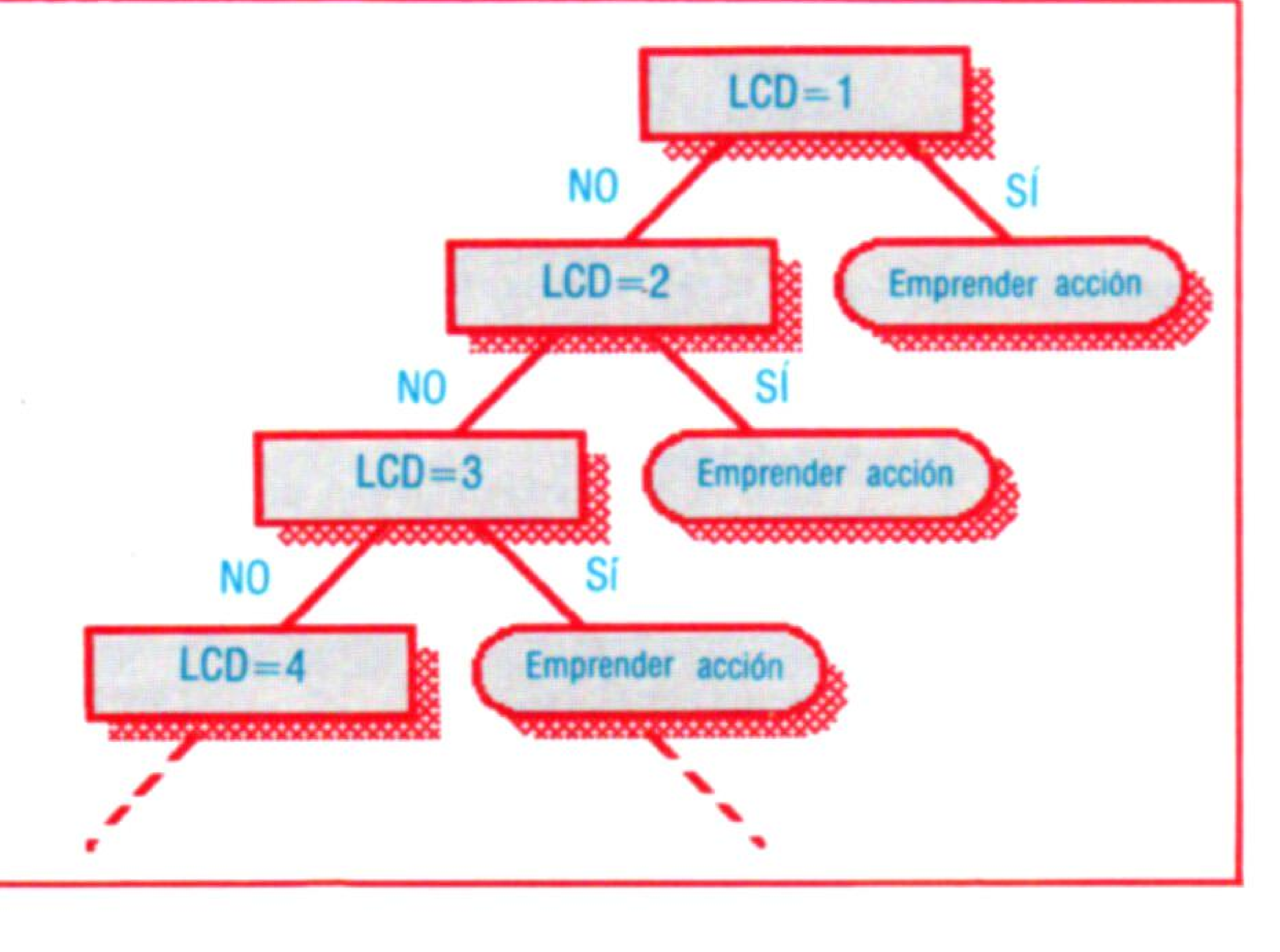
Es importante comprender que si usted tiene intención de aplicar estas ideas en un juego de aventuras propio debe planificar su árbol de modo que no se impriman mensajes en la pantalla con demasiada frecuencia. Aunque considerara acertada la idea de hacer aparecer gran cantidad de mensajes, éstos se imprimen con excesiva frecuencia pueden dificultar su progreso a través del juego.

Como regla general, puede dividir las acciones de los personajes en tres niveles. El nivel superior implica acciones tales como dar o entregar un objeto y exige que se imprima un mensaje, si es que usted está presente, informándole sobre lo sucedido. El segundo nivel implica aquellas acciones que no inciden necesariamente en la trama del juego pero que, no obstante, podrían visualizar mensajes para contribuir a la creación de una cierta atmósfera. También se podría visualizar un mensaje si usted entrara **Examine a un personaje** o palabras con esa intención. El tercer nivel (alterar el valor de la bandera de "humor" en nuestro propio programa, p. ej.) jamás tiene como consecuencia la impresión de un mensaje.

Vale la pena tener presente esta idea de "niveles" cuando usted programa su propio juego; puede asignarle un valor a cada acción que determine su nivel y luego tener una sola subrutina para ocuparse de la impresión de mensajes. Tal rutina comprobaría primero si usted está presente, comprobaría el nivel de la acción si lo estuviera, y decidiría si imprimir o no un mensaje.

Limitaciones de la bifurcación

La utilización de una estructura arborescente en la que cada nudo sólo pueda bifurcarse en dos direcciones puede imponernos serias limitaciones al comprobar las condiciones para una cantidad de valores diferentes



Volviendo a nuestra rutina de manipulación de objetos, intente primero suprimir la línea 5010. Ello tendrá el efecto de asegurar que se impriman mensajes para todos los personajes, independientemente si se hallan en la misma habitación que el jugador o no. Usted verá que, durante la mayor parte del tiempo, los personajes se concentran en hacerse con sus "propios" objetos y bebérselos, que es lo que deseamos si hemos de mantener un grado de realismo. No obstante, puede experimentar con esto alterando los valores por defecto de las sentencias de datos en las líneas 6030 y 6040. Por ejemplo, si altera los inventarios de Lola Fiestas, Pepe Viñas y Gina Fizz a 9, 12 y 8 respectivamente, y luego ejecuta (RUN) el programa, verá que las acciones de la sala de tertulia siguen un curso bastante diferente. Pruebe de "regular" a los otros personajes de este modo y vea lo que sucede.

Además de cambiar la línea 5010 como indicamos arriba, usted también hallará que "regular" a sus personajes le resulta más fácil saltándose las líneas 560 a 580 de modo que cada personaje se procese mediante cada llamada a la rutina manipuladora, en vez de dejarlos esperando hasta que sus banderas se reduzcan a cero.

Para hacer esto, simplemente edite la línea 550 de modo que rece:

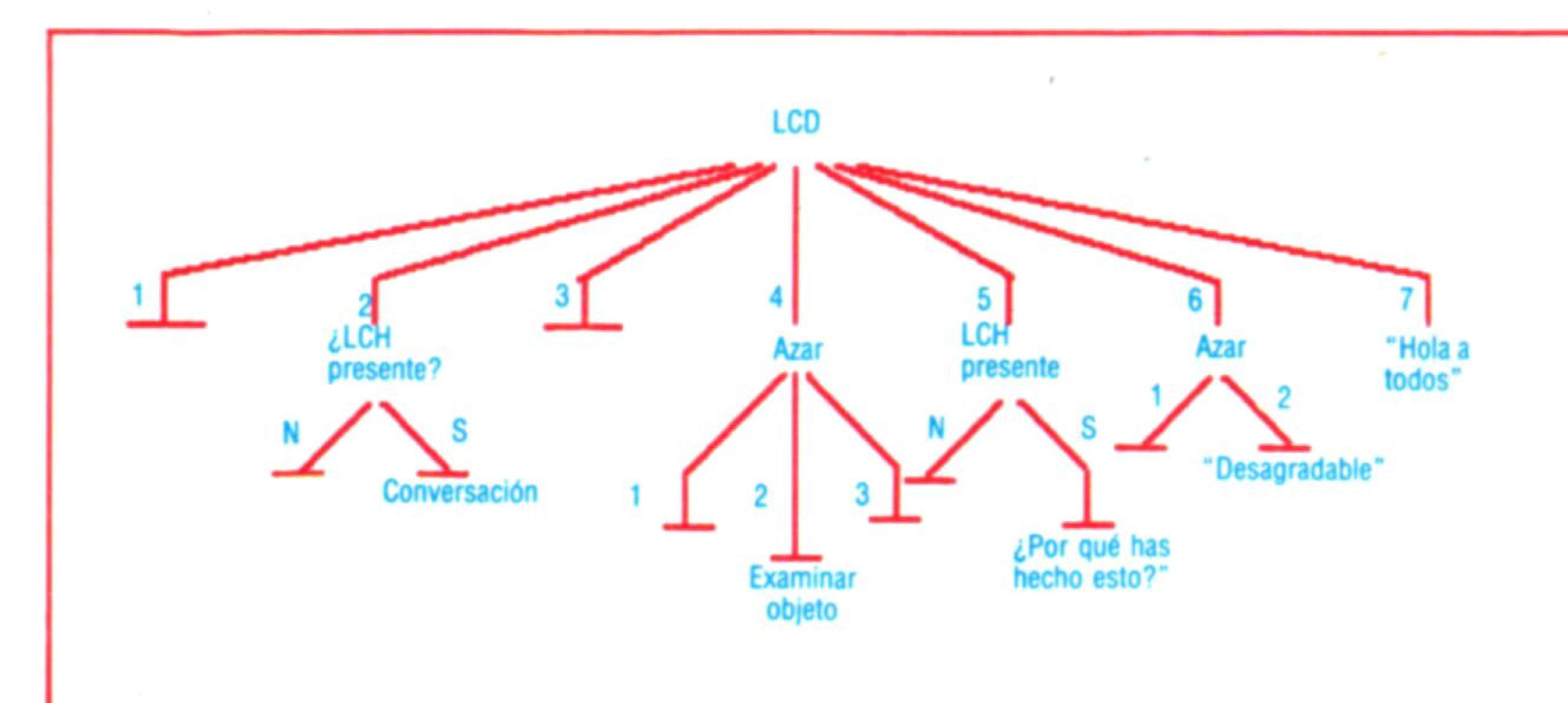
```
550 FOR c=1 TO 6:GOTO 590
```

Ahora verá que las acciones de cada personaje se visualizan una después de otra. Recuerde, si decide que el programa principal no está actualizando a los personajes con suficiente rapidez, que siempre puede alterar los valores para las banderas "manipular" en las líneas 6030 y 6040. (Si ha olvidado exactamente a qué factor alude cada elemento de la matriz c, remítase a p. 1972.)

Pasemos ahora a la interacción. En la línea 190 ya hemos DIMensionado la matriz t para retener los datos de tres árboles, cada uno con hasta 25 nudos de elección. Tal vez encontremos que hay que alterar este valor, pero, aún más importante, tal vez encontremos también que la estructura de árbol que hemos utilizado para tratar con los objetos no es lo suficientemente poderosa para la interacción. Para ver por qué esto es así, observemos el árbol de objetos de 2055. Verá que una de las características de diseño fundamentales de este árbol es que cada nudo se bifurque solamente en dos únicas direcciones.

Esta configuración, obviamente, es muy conveniente, puesto que cada una de las condiciones que estamos comprobando en los nudos de elección sólo tiene dos posibles valores: falso o verdadero. Supongamos, no obstante, que las condiciones hubieran de tener más de dos posibles valores. Si quisiéramos comprobar el valor de la bandera LCD (c$(n,9)), por ejemplo, que puede tener siete posibles valores (ver p. 1974) utilizando un árbol como el de p. 2076, necesitaríamos algo como la estructura que se mostraba en el primer diagrama. Es obvio que con esto es bastante incómodo, porque sería mucho más fácil si pudiéramos recurrir a una estructura como la que anteriormente mostraba el diagrama 2.

En realidad, podemos hacerlo con bastante facilidad, empleando aún la matriz t para retener datos



Opción múltiple

Este árbol muestra cómo el programa podría procesar la bandera de Código de Última Instrucción, LCC (c$(c,9)) utilizando una estructura en la cual cada nudo pueda bifurcarse en más de dos direcciones. Este método permite implementar en nuestro programa diseños arborescentes más compactos

para nuestro nuevo árbol. Nuestro primer árbol utilizaba t(n,n,1) y t(n,n,2), que retenían los números de nudo nuevos a los que habría de bifurcarse según el valor retenido en la matriz c fuera uno o dos. No obstante, podríamos utilizar t(n,n,1) para retener un número de nudo *base* y a t(n,n,2) para retener un *offset*. De modo que, por ejemplo, en el segundo diagrama el número de nudo uno tendría el valor dos leído en el elemento de la matriz t(n,1,1) y el valor de LCD menos uno leído en t(n,1,2). Aplicando este procedimiento podríamos recorrer el árbol mediante la fórmula:

```
Nuevo codigo = t(numero de arbol, numero de nudo actual, 2) + t(numero de arbol, numero de nudo actual,1)
```

Éste es el método que emplearemos para nuestro árbol de interacción. Tiene una o dos limitaciones, pero permite diseñar una estructura muchísimo más compacta para el manipulador de personajes. En el tercer diagrama se ilustra una estructura arborescente provisional empleando este procedimiento. Primero, comprueba la fortaleza de los personajes, porque si un personaje está muerto o inconsciente, es obvio que no va a hacer nada como no sea actuar como centro de atención de los demás. El programa decide luego si mover o no un personaje, comprobando (y actualizando si fuera necesario) la bandera "mover" retenida en c$(n,11). Por último, se elige uno de tres subárboles: interacción con personajes, conciencia de objeto e informes de actividad.

Complementos al BASIC

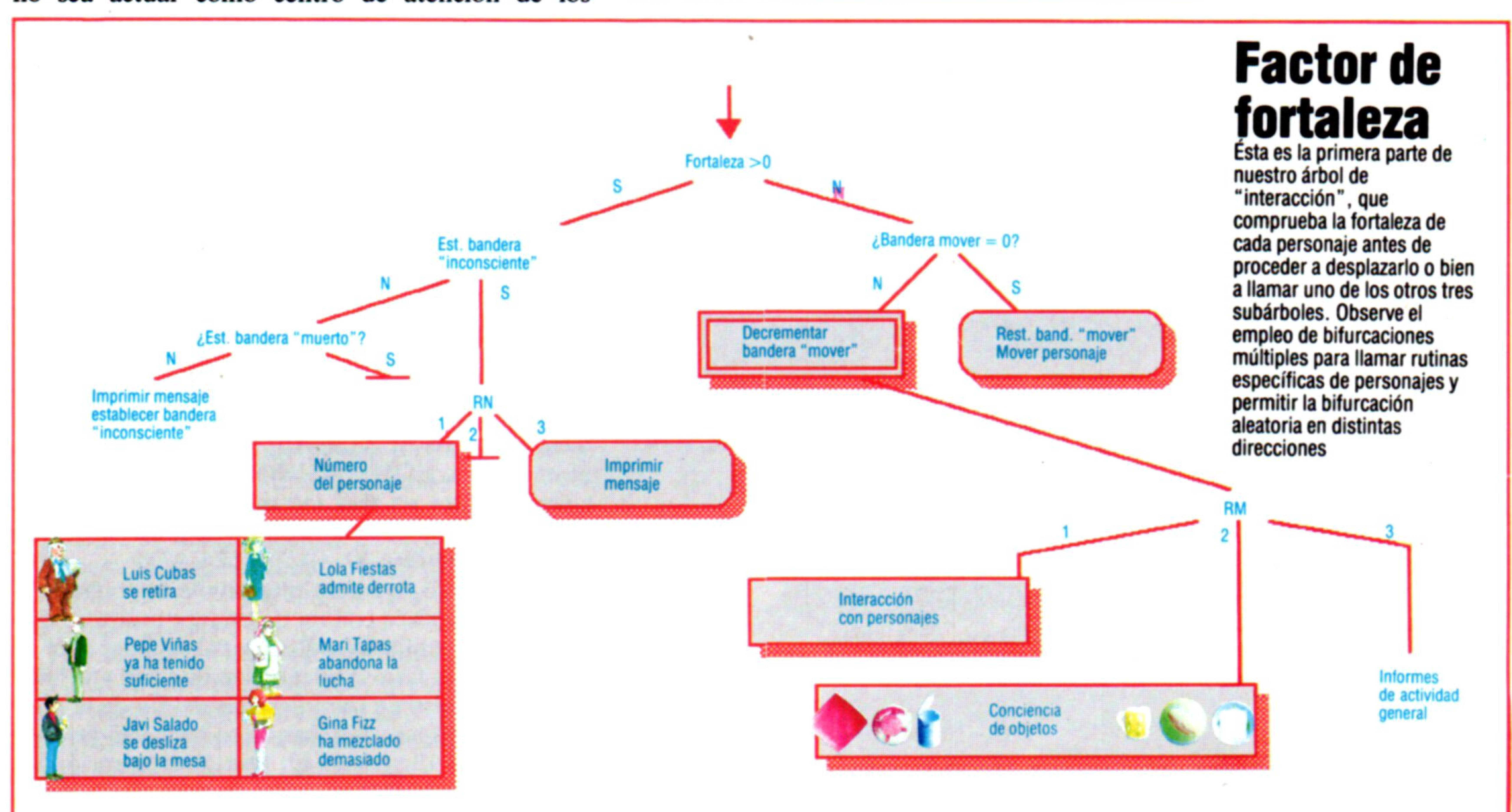
En el listado que ofrecimos en el capítulo anterior se deben introducir las siguientes modificaciones y adiciones:

Spectrum:

```
4180 q=INT(RND*2)+1:RETURN
5080 IF n=23 THEN GOSUB 2540:GOTO 5040
5085 GOSUB 2640:GOTO 5040
5090 RESTORE 9900:FOR e=1 TO (n-23):READ h:
     NEXT e:GOTO h
9900 DATA 5100,5130,5160,5180,5210,5240,
     5260,5270,5280,5300,5310,5330,5340,
     5360,5370,5430
```

BBC Micro:

```
4180 q=RND(2):RETURN
```



Factor de fortaleza

Ésta es la primera parte de nuestro árbol de "interacción", que comprueba la fortaleza de cada personaje antes de proceder a desplazarlo o bien a llamar uno de los otros tres subárboles. Observe el empleo de bifurcaciones múltiples para llamar rutinas específicas de personajes y permitir la bifurcación aleatoria en distintas direcciones

Caroline Clayton

Conclusión del proyecto

En este capítulo final realizaremos dos ampliaciones opcionales al diseño original antes de dar por terminado nuestro tester

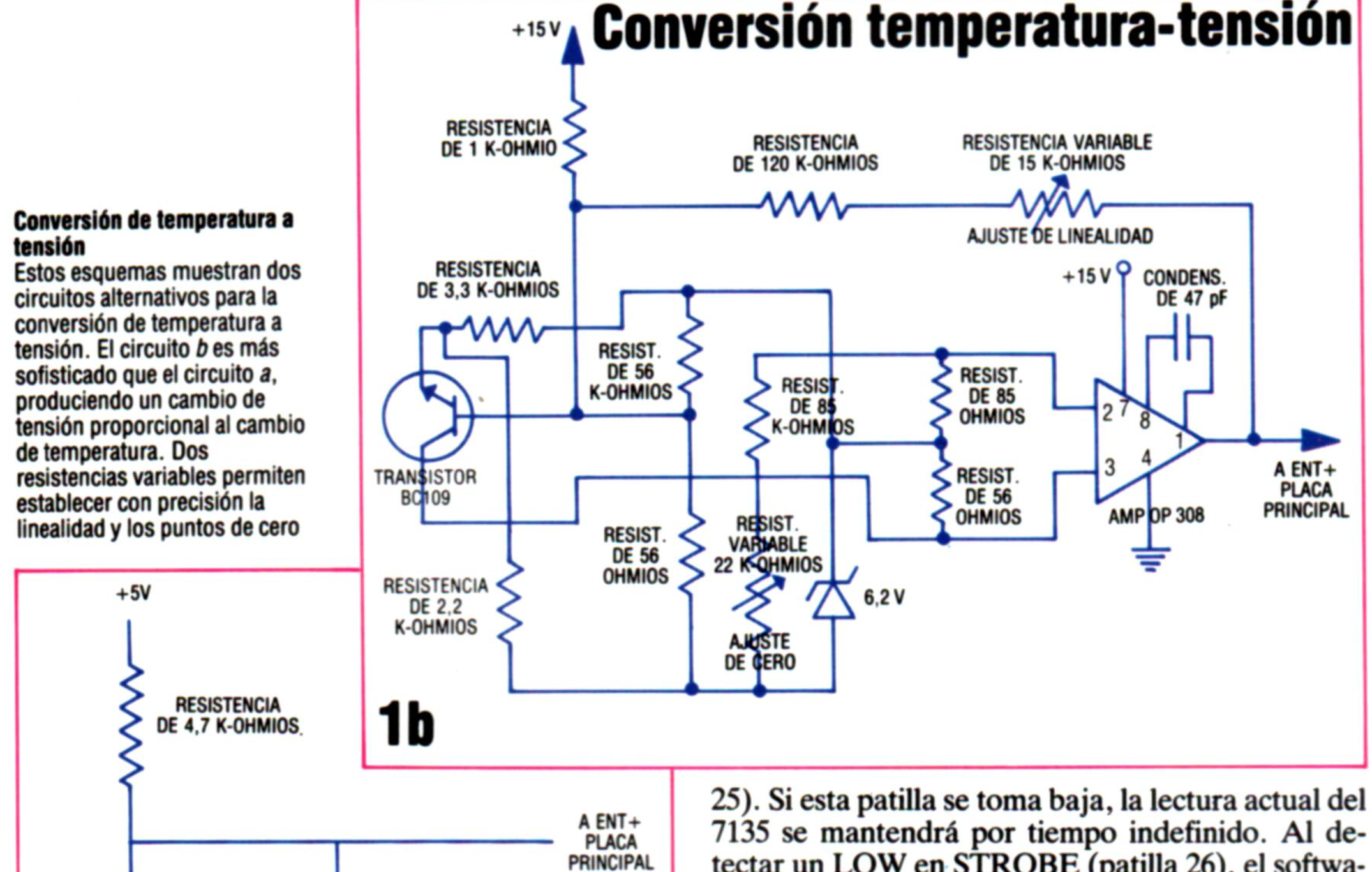
La forma más fácil de medir temperatura con un DVM es comprar una punta de prueba de temperatura ya hecha que produce una salida de tensión proporcional a la temperatura medida. Sin embargo, por lo general no son baratas, y es fácil construir dispositivos similares con un coste bastante menor. En el esquema 1 ofrecemos dos circuitos posibles. El primero, el más sencillo, produce una salida de tensión que aumenta en 10 mv por °C, pero a la salida habrá que restarle tensión constante para obtener la lectura necesaria. Si se conecta el DVM a un ordenador, no habrá ningún problema, porque la resta se puede realizar mediante software. La escala normal es de −10° a +100 °C.

Conexión en interface

El chip convertidor A/D 7135 se ha diseñado específicamente para que la conexión en interface a un microprocesador sea lo más sencilla posible, y esto se puede hacer de numerosas maneras. En primer lugar, las salidas digitales del chip están en BCD (decimal codificado en binario). Las salidas de "dígito" separadas se vuelven verdaderas para indicar para qué dígito es válido el dato BCD. Hay una salida STROBE (de sentido negativo) que se puede utilizar para enclavar los datos BCD en un UART. El impulso STROBE se vuelve momentáneamente negativo una vez para cada dígito (cinco veces durante cada ciclo de medición). El impulso se produce en el centro del impulso TRUE de activación de cada dígito.

Si el software para lectura de datos por ordenador está escrito en BASIC u otro lenguaje de alto nivel, sería ventajoso enviar una salida desde el ordenador a la patilla RUN/HOLD del 7135 (patilla 25). Si esta patilla se toma baja, la lectura actual del 7135 se mantendrá por tiempo indefinido. Al detectar un LOW en STROBE (patilla 26), el software enviará un LOW a RUN/HOLD y puede tardar cuanto quiera en leer los datos BCD, utilizando como impulsos de reloj los impulsos de activación de dígitos (patillas 20, 19, 18, 17 y 12).

Sin embargo, la forma más simple de conectar el DVM en interface con un ordenador utiliza la salida BUSY (patilla 21). Esta señal se hace alta al principio de la fase de integración de señal y se hace baja un impulso de reloj después del fin de la integración de referencia (cuando se enclavan las salidas de datos digitales). La fase de integración de señal emplea 10 000 impulsos de reloj (y BUSY se pone bajo al final del primer impulso después de



Conversión de temperatura a tensión
Estos esquemas muestran dos circuitos alternativos para la conversión de temperatura a tensión. El circuito *b* es más sofisticado que el circuito *a*, produciendo un cambio de tensión proporcional al cambio de temperatura. Dos resistencias variables permiten establecer con precisión la linealidad y los puntos de cero



pasar por cero). Relacionando con ADN a CLOCK y BUSY y contando la cantidad de impulsos transmitidos al ordenador mediante la puerta AND, se podrá, por tanto, obtener la tensión medida simplemente restando la cantidad de 10 001 por software.

La rutina para contar los impulsos habrá de escribirse en lenguaje máquina, porque el BASIC sería demasiado lento. A una frecuencia de reloj de 125 KHz, el impulso será positivo para 4 µS.

Una alternativa aún más simple para contar los impulsos sería una rutina sencilla que esperara a que BUSY se pusidera alto y luego efectuara un bucle hasta que BUSY se pusiera bajo, sumándole uno a COUNT cada vez que se realizara el bucle. Para que esta técnica funcione, habrá de conocerse con exactitud la velocidad de la CPU del ordenador y habrá de calcularse la cantidad exacta de ciclos de máquina requeridos para cada instrucción en código máquina. Asimismo, también será necesario haber medido con exactitud la frecuencia de reloj del DVM. Esto permitirá medir la duración de BUSY con bastante precisión, restando 10 001 veces el período de CLOCK para obtener la duración de la fase de integración de referencia, y dividir este resultado por el período de CLOCK para obtener la cantidad de impulsos CLOCK en la integración de referencia. Cada impulso contado corresponde entonces a 0,0001 V.

Las señales BUSY y CLOCK del chip 7135 se pueden relacionar fácilmente con una AND utilizando un chip 74LS00, que posee cuatro puertas NAND. La figura 2 muestra las conexiones de patillas necesarias para producir la señal BUSY-.CLOCK. Este chip también se puede utilizar para proporcionar únicamente una señal de BUSY a través de un tampón si se desea adoptar el enfoque simple para medir la duración de la fase de integración de referencia que acabamos de describir.

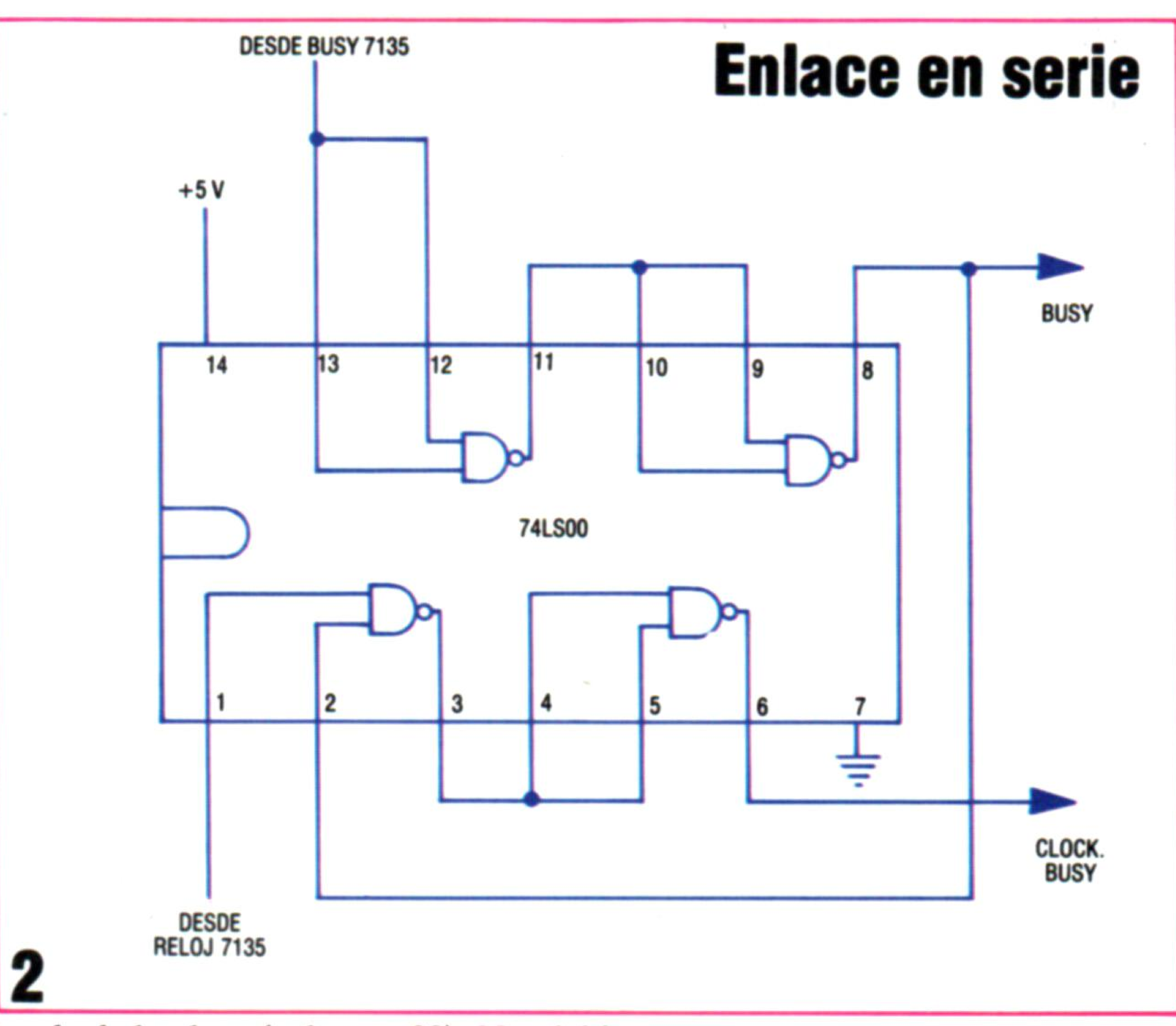
Modificaciones de diseño

Nuestro prototipo para el DVM utilizó un tipo de LED distinto para el dígito 5, con un signo "más" y "menos" separado. Debido a la dificultad para obtener un LED adecuado, modificamos el diseño para permitir el empleo de un LED de siete segmentos común para el dígito 5 (utilizando el segmento *g*, la barra cruzada, como signo menos).

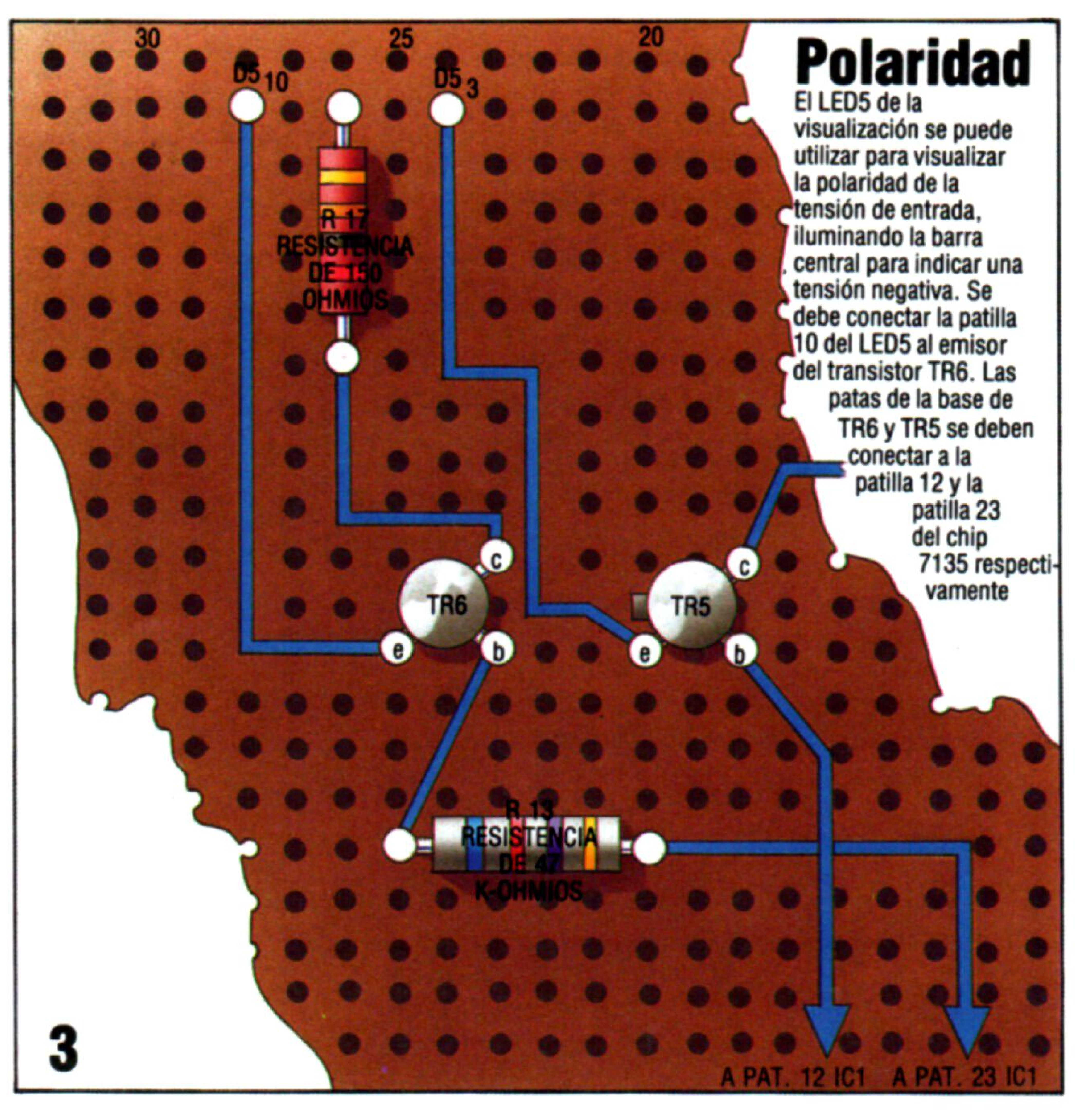
Ello supone la introducción de algunas ligeras modificaciones en el circuito publicado anteriormente. Puesto que el LED de siete segmentos tiene sólo un ánodo común para los siete segmentos, el signo menos sólo puede estar encendido cuando el dígito 5 está totalmente iluminado (en la disposición anterior había un ánodo separado para el signo). Para utilizar el LED 5 de modo que muestre un signo menos, en el diagrama de trazado correspondiente se deben efectuar las siguientes modificaciones. TR5 queda como estaba; el colector va a la fuente de +5 V y el emisor va al ánodo común del LED5 (patilla 3).

TR6 se conecta del siguiente modo: su base va a la patilla 23 (polaridad) de IC1, su emisor se conecta al cátodo del segmento *g* del LED5 (patilla 10) y su colector se conecta a la masa digital a través de una resistencia de 150 ohmios. En el diagrama de construcción, la resistencia limitadora de corriente desde el colector de TR6 a masa se lleva hasta el borde de la placa (columna 30). No olvide conectar este punto a la masa digital.

Y así termina nuestro proyecto del tester. Le hemos ofrecido los detalles completos para construir un medidor básico de 2 V de sensibilidad, junto con detalles bastante completos para numerosos accesorios y una forma sencilla de conectar el medidor en interface a cualquier ordenador. Queda en sus manos decidir el estilo de la caja para alojar la unidad.



Enlace en serie
Las señales provenientes del chip convertidor A/D 7135 se pueden conectar en interface a través de un chip 74LS00 para producir una salida CLOCK. BUSY. Conectando esta salida a un ordenador y contando por software la cantidad de impulsos recibidos, se puede calcular la entrada de tensión en el voltímetro



Polaridad
El LED5 de la visualización se puede utilizar para visualizar la polaridad de la tensión de entrada, iluminando la barra central para indicar una tensión negativa. Se debe conectar la patilla 10 del LED5 al emisor del transistor TR6. Las patas de la base de TR6 y TR5 se deben conectar a la patilla 12 y la patilla 23 del chip 7135 respectivamente

Traducir fórmulas

Ahora nos dedicaremos a las rutinas que permiten que el programa entre fórmulas matemáticas y las evalúe

Cuando se trabaja con una hoja electrónica, una de las principales consideraciones es que, tal como ocurre con tantas otras aplicaciones, usted y el ordenador poseen diferentes métodos de efectuar cálculos. Supongamos, por ejemplo, que entrara en la celda A5 la fórmula (A1+A2)* (A3−A4). Según el conjunto de reglas que nos han enseñado, primero debemos evaluar el contenido de cada par de paréntesis por separado y después multiplicar los resultados entre sí. Si quisiéramos que el ordenador efectuara este cálculo, habríamos de indicarle que hiciera lo siguiente:

1. Tomar los valores de las celdas A1 y A2.
2. Sumarlos entre sí y almacenar el resultado temporalmente.
3. Tomar los valores de las celdas A3 y A4.
4. Restarle a A4 el valor de A3.
5. Multiplicar el resultado por el resultado de A1 más A2.
6. Almacenar el resultado en la celda A5.

Usted, probablemente, podría escribir un programa para llevar a cabo esta función, pero no sería de mucha utilidad si tuviera que evaluar muchas fórmulas difefrentes. En consecuencia, lo que necesita es una rutina que tome cualquier fórmula y la evalúe por el orden correcto.

El problema de escribir expresiones en la forma habitual (que se conoce como notación de *infijos*) es que un ordenador no las evaluará por el mismo orden. En cambio, para escribir expresiones utilizamos el sistema de notación polaca inversa (*reverse Polish notation*: RPN).

La conversión y evaluación de fórmulas en nuestra hoja electrónica se puede dividir en tres pasos distintos: el paso 1 es el auténtico proceso de convertir la serie de infijos a RPN; el paso 2 consiste en comprobar que la serie RPN producida esté bien constituida (sea sintácticamente correcta); el paso 3 es la evaluación propiamente dicha de la fórmula.

Los operadores que permite este programa son +, −, *, / y ↑. Si usted utiliza en una fórmula el signo − para indicar un valor negativo, el programa exige el carácter & para diferenciar éste del signo menos normal. Para llevar a cabo la conversión de infijos a RPN, necesitamos asignarle a cada uno de los operadores valores de precedencia. Esto determina el orden por el cual se trabaja con los operadores. La división y la multiplicación, por ejemplo, se efectúan antes de la suma y la resta:

El diagrama de flujo muestra el algoritmo para convertir una serie de infijos, I$, a una serie de notación polaca inversa, P$, y corresponde a la subrutina de la línea 4000 del programa. A partir de este diagrama usted podrá hacerse una idea de la forma en que trabaja la rutina. Los operadores admitidos por la rutina de conversión son los que se mencionan en el texto, mientras que los operandos permitidos son los nombres de las celdas y constantes reales. El principal problema de emplear el BASIC para el programa es que hemos de simular una pila LIFO. Esta función se lleva a cabo mediante el empleo de la matriz ST$() y la variable SP como señalador de la pila, utilizándose la primera para apilar los elementos de P$ durante el proceso de conversión



Operador	Valor de precedencia
=	0
(	1
+−)	2
*/&	3
↑	4

En la versión final del programa de hoja electrónica, la rutina para convertir fórmulas de infijos a notación polaca inversa se acomodará así. Después de que entre una fórmula que relacione matemáticamente varias celdas, la serie de infijos resultante se coloca en una matriz, F$(), que almacena las fórmulas para cada celda. Luego se borra el elemento correspondiente de una segunda matriz, PS$(), la cual se utiliza para retener la correspondiente serie RPN.

Cuando seleccione la función calcular, el programa recorrerá la matriz PS$(), tomando cada serie polaca inversa de una en una y preparándola para la evaluación. Si desde la última función calcular se hubiera entrado en la hoja electrónica una nueva fórmula para una celda determinada, la entrada RPN para esa celda estaría vacía. Por tanto, la rutina de conversión que presentamos aquí es llamada antes de preparar la fórmula de la celda para su evaluación. Utilizando esta disposición, podemos retener versiones tanto de infijos como el RPN para cada fórmula de la hoja, usando la primera para visualizar en la pantalla y la segunda para la evaluación por ordenador.

Conversión de series

Complementos al BASIC

BBC Micro:

Introducir estas modificaciones en la versión para el C64:

Omitir la línea 50

```
4240 IF ST$(SP)="(" OR
     ST$(SP)=")" THEN
     PRINT TAB(0,22);"
     ERROR EN LA FORMULA-
     ":RETURN
```

Amstrad CPC 464/664:

Introducir estas modificaciones en la versión para el C64:

Omitir la línea 50

```
4240 IF ST$(SP)="(" OR
     ST$(SP)=")" THEN LO-
     CATE 1,22:PRINT "
     ERROR EN LA FORMULA-
     ":RETURN
```

La línea 3140 se debe renumerar 3160

Commodore 64:

```
50 CD$=CHR$(17):CU$=CHR$(145):CR$=CHR$(29):CL$
   =CHR$(157):CO$=CHR$(19)
```

Adición a la subrutina de matrices

```
3140 DIM F$(225):DIM PS$(255)
```

Rutina de conversión RPN

```
4000 REM ********************
4001 REM * CONVERSION DE SERIE NORMAL *
4002 REM * A POLACA INVERSA
4003 REM ********************
4005 FOR K=1 TO 20:ST$(K)=" ":NEXT K
4006 LET P=0:LET SP=0:P$=""
4010 LET I$=C$
4015 FOR K=1 TO LEN(I$)-1
4017 J$=MID$(I$,K,1):K$=MID$(I$,K+1,1)
4020 IF J$="(" AND K$="-" THEN
     I$=MID$(I$,1,K)+"&"+MID$(I$,K+2)
4025 NEXT K
4030 IF P<=LEN(I$) THEN 4100
4040 REM *** PILA VACIA *****
4050 IF SP=0 THEN PS$((J-1)*15+I)=P$:GOSUB
     4700:RETURN
4060 IF ST$(SP)="(" OR ST$(SP)=")" THEN 4080
4070 LET P$=P$+ST$(SP)
4080 LET SP=SP-1
4090 GOTO 4050
4095 STOP
4100 LET P=P+1
4110 LET S$=MID$(I$,P,1)
4120 IF S$="+" OR S$="-" OR S$="*" OR S$="/" THEN
     4200
4130 IF S$="^" OR S$="(" OR S$=")" OR S$="&" THEN
     4200
4140 LET P$=P$+S$
4150 GOTO 4030: REM CONTINUAR HASTA EL FINAL DE LA
     SERIE
4200 IF SP=0 THEN 4400
4210 IF S$="(" THEN 4400
4220 GOSUB 4500:REM ** PRECEDENCIA **
4230 IF PS> PT THEN 4300:REM PRECEDENCIA
4240 IF ST$(SP)="("OR ST$(SP)=")" THEN GOSUB
     1950:PRINT" ERROR EN LA FORMULA ":
     RETURN
4250 LET P$=P$+ ST$(SP):ST$(SP)="":LET SP=
     SP-1
4260 IF SP> 0 THEN 4220
4270 SP=SP+1
4280 LET ST$(SP)=S$
4290 GOTO 4030: REM CONTINUAR HASTA FINAL DE LA
     SERIE
4300 IF ST$(SP)="(" AND S$=")" THEN LET ST$(SP)=""
     :SP=SP-1 :GOTO 4030
4400 LET SP=SP+1
4410 LET ST$(SP)=S$
4420 GOTO 4030
4500 REM *** EVALUACION DE PRECEDENCIA ***
4510 LET T$=S$:GOSUB 4600
4520 LET PS=T
4530 LET T$=ST$(SP):GOSUB 4600
4540 LET PT=T
4550 RETURN
4600 IF T$="=" THEN T=0:RETURN
4605 IF T$="(" THEN T=1:RETURN
4610 IF T$="+" OR T$="-" OR T$=")" THEN T=2:
     RETURN
4620 IF T$="*" OR T$="/" OR T$="&" THEN T=3:
     RETURN
4630 IF T$="^" THEN T=4:RETURN
4650 T=0:RETURN
```

Sinclair Spectrum:

Adición a la subrutina de matrices

```
3150 DIM F$(255,20):DIM H$(255,20):RETURN
```

Rutina de conversión RPN

```
4000 > REM ********************
4001 REM * CONV. DE SERIE NORMAL *
4002 REM * A POLACA INVERSA      *
4003 REM ********************
4005 DIM S$(20)
4006 LET P=0:LET SP=0: LET P$=""
4010 LET I$=C$
4011 FOR Z=1 TO LEN (I$)
4012 IF I$(Z)=" " THEN GO TO 4014
4013 NEXT Z
4014 LET I$=I$( TO Z-1)
4015 FOR K=1 TO LEN (I$)-1
4017 LET J$=I$(K): LET K$=I$(K+1)
4020 IF J$="(" AND K$="-" THEN LET I$=I$
     (1 TO K)+"&"+I$(K+2 TO)
4025 NEXT K
4030 IF P< LEN(I$) THEN GO TO 4100
4040 REM **** PILA VACIA ****
4050 IF SP=0 THEN LET H$((J-1)*15+I,
     1 TO)=P$: GO SUB 4700:RETURN
4060 IF S$(SP)="(" OR S$=")" THEN GO TO
     4080
4070 LET P$=P$+S$(SP)
4080 LET SP=SP-1
4090 GO TO 4050
4095 STOP
4100 LET P=P+1
4110 LET O$=I$(P)
4120 IF O$="+" OR S$="-" OR O$="*" OR O$
     ="/" THEN GO TO 4200
4125 IF O$="^" OR O$="(" OR O$=")" OR O$
     ="&" THEN GO TO 4200
4140 LET P$=P$+O$
4150 GOTO 4030
4200 IF SP=0 THEN GO TO 4400
4210 IF O$="(" THEN GO TO 4400
4220 GO SUB 4500: REM PRECEDENCIA
4230 IF PS> PT THEN GO TO 4300
4240 IF S$(SP)="("OR S$(SP)=")" THEN PRINT
     AT 20,0;"ERROR EN LA FORMULA ":
     RETURN
4250 LET P$=P$+ S$(SP): LET SP=SP-1
4260 IF SP> 0 THEN GO TO 4220
4270 SP=SP+1
4280 LET S$(SP)=O$
4290 FO TO 4030: REM CONTINUAR HASTA FIN
     DE LA SERIE
4300 IF S$(SP)="(" AND O$=")" THEN LET
     S$(SP)="" : LET SP=SP-1: GO TO 4030
4400 LET SP=SP+1
4410 LET S$(SP)=O$
4420 GO TO 4030
4500 REM * EVALUACION DE PRECEDENCIA *
4510 LET T$=S$:GO SUB 4600
4520 LET PS=T
4530 LET T$=S$(SP):GO$SUB 4600
4540 LET PT=T
4550 RETURN
4600 IF T$="=" THEN LET T=0: RETURN
4610 IF T$="(" THEN LET T=1: RETURN
4620 IF T$="+" OR T$="-" OR T$=")" THEN
     T=2: RETURN
4630 IF T$="*" OR T$="/" OR T$="&" THEN
     LET T=3: RETURN
4640 IF T$="^" THEN LET T=4: RETURN
4650 T=0:RETURN
```

En la era del jet
La Epson SQ-2000 es una impresora de chorro de tinta de alta calidad que incorpora una amplia gama de estilos de impresión e impresión de gráficos en alta resolución. Configura el nuevo sistema de Epson de cartucho de tinta y limpieza automática

LA JET SET
LA JET SET

La Epson SQ-2000 es una excelente representante de las impresoras de chorro de tinta ("ink jet printer")



Espacio para maniobrar
La Epson SQ-2000 no incluye interface para ordenador. En cambio, en la parte posterior de la máquina se puede instalar una de tres interfaces estándares: Centronics, RS232C e IEEE. Aquí también se puede añadir un tampón opcional de 32 K y tipos adicionales en ROM

LA JET SET
LA JET SET



Crispin Thomas

En los últimos años, el precio medio de las impresoras matriciales de gran calidad se ha reducido rápidamente hasta ponerse al alcance de muchos usuarios de ordenadores personales. La nueva gama de impresoras Epson parece estar pronta a hacer lo mismo con la tecnología de chorro de tinta. La SQ-2000, que describimos aquí, es la que actualmente está en el mercado, pero es probable que Epson introduzca otra impresora de chorro de tinta A4 unidireccional (que se llamará HS-80). Esto pondrá a las impresoras de chorro de tinta al alcance de los usuarios de ordenadores personales.

Al igual que sus parientes de la familia (la Epson FX y la gama de impresoras matriciales MX), la SQ-2000 viene alojada en una atractiva carcasa de plástico color crema. Cuando la máquina descansa sobre un escritorio, los pulsadores de control, el interruptor de red y la cámara de cartuchos de tinta quedan fácilmente accesibles. Dado que está diseñada básicamente para pequeñas empresas, la SQ-2000 posee un carro ancho para dar cabida a hojas especiales de contabilidad y a hojas electrónicas. Deliberadamente, Epson ha hecho que su nueva impresora sea lo más flexible posible. El papel se puede alimentar mediante el rodillo portapapel normal y el método de guías de hoja, pero también se pueden adquirir como extras alimentadores de papel de hojas cortadas o de tractor continuo. La unidad básica no tiene incorporada interface para ordenador, pero en la parte posterior de la máquina hay una ranura para enchufar un cartucho de interface. Es posible escoger entre tres tipos de interface estándares distintos: en paralelo Centronics, en serie RS232C o IEEE 488, según el tiempo de ordenador con el que se usará la máquina. El cartucho de interface se instala fácilmente empujándolo en su sitio y asegurándolo con un par de tornillos. Las interfaces estándares tienen un tampón de entrada de 2 K en donde almacenar temporalmente los datos que entran procedentes del ordenador. Un buffer interno de doble impresión permite la decodificación e impresión simultánea de los datos tomados del tampón de entrada, y hay disponible un tampón de caracteres de carga inferior de 5 K para almacenar caracteres definidos por el usuario.

Las impresoras de chorro de tinta presentan muchas ventajas sobre las matriciales y las margarita. Combinan la calidad de impresión de las impresoras margarita con la velocidad y la flexibilidad de las impresoras matriciales, y sin el nivel de ruido. De hecho, la SQ-2000 es considerablemente más silenciosa que ninguna otra impresora matricial o margarita, lo que representa una importante ventaja. Sin embargo, las impresoras de chorro de tinta han experimentado problemas que sólo recientemente han podido solucionarse: las cámaras de tinta tendían a quedarse bloqueadas y las máquinas solían ser difíciles de manejar. Epson afirma que su nuevo sistema de entrega de tinta evita estos

problemas. En este sistema, la tinta se suministra desde un cartucho de tinta sellado que consta de tres cámaras, que contienen una tinta de fórmula especial, un líquido de limpieza y un espacio de almacenamiento para los productos de desecho. El cartucho está diseñado para imprimir más de tres millones de caracteres antes de su sustitución.

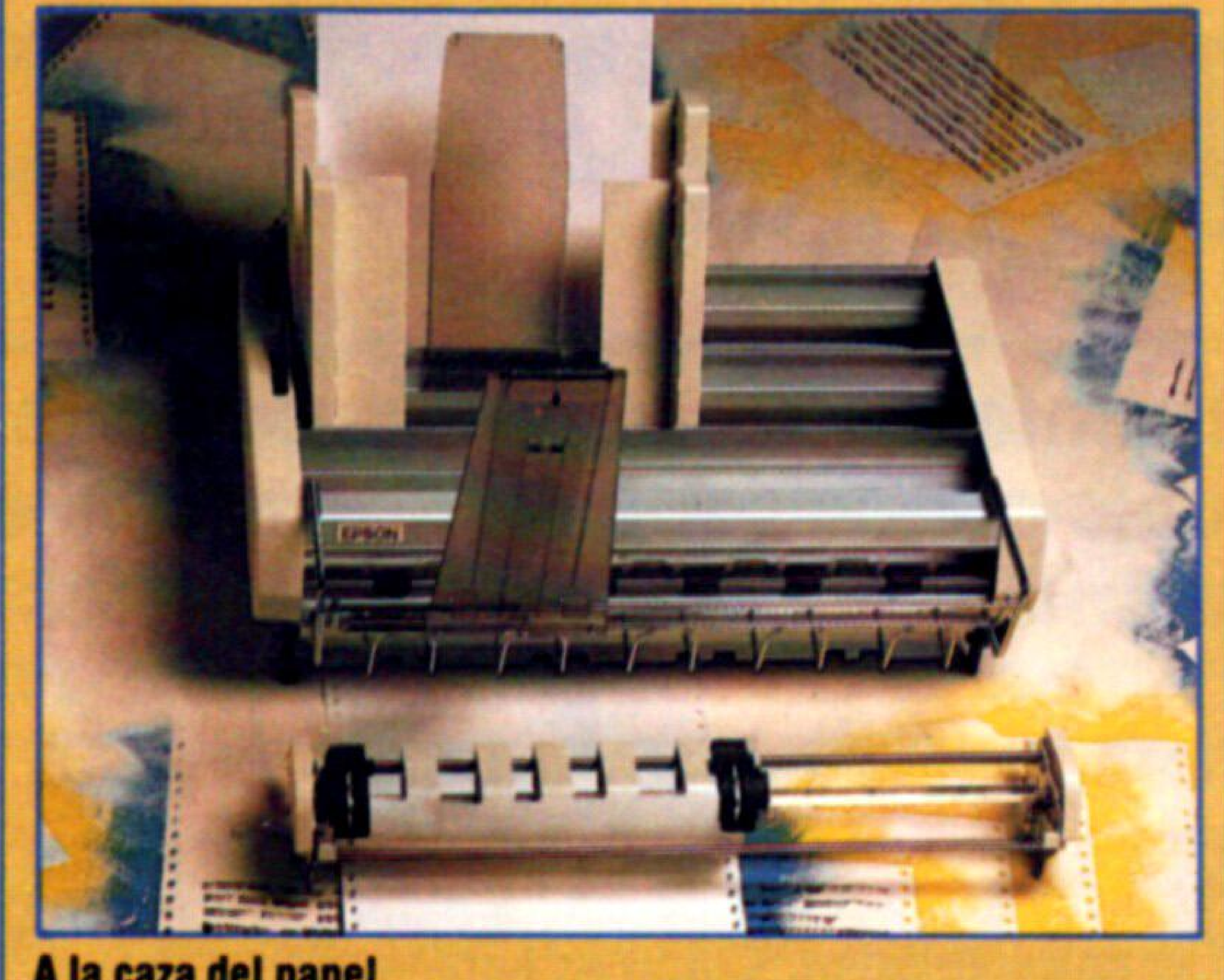
La razón por la cual Epson alega haber mejorado la fiabilidad es que el sistema de suministro de tinta se limpia de forma automática (operación que lleva apenas unos pocos segundos) cada vez que se enciende o se apaga la máquina, y periódicamente mientras la máquina se halla en funcionamiento. Esta operación de limpieza también se puede desencadenar manualmente manteniendo apretado un pulsador situado en el panel de control delantero. Los otros controles de este panel incluyen los conocidos pulsadores *on-line, line* y *form feed,* así como un botón de alimentación de hojas que permite cargar un hoja única desde un sujetapapel. Tres luces indicadoras de color verde señalan que la máquina está en marcha, si la máquina está o no en línea y si está lista para recibir datos desde el ordenador anfitrión. Dos luces rojas indican que se está acabando la tinta o que se ha agotado el papel.

Una de las características más impresionantes de la SQ-2000 es su gama de estilos de impresión. Hay dos modalidades básicas disponibles: borrador y calidad casi de impresión *(near letter quality*: NLQ). La modalidad de borrador es más rápida que la NLQ, pero posee una pérdida proporcional de calidad de impresión. En cada modalidad se pueden seleccionar tipos Pica, Elite y Roman en formas ampliada, condensada, cursiva, subrayada, realzada, proporcional e índice/subíndice.

Al igual que los producidos mediante una impresora matricial, los caracteres de una impresora de chorro de tinta se forman a partir de puntos individuales, permitiendo la flexibilidad de formas de caracteres programables. No obstante, debido a que los puntos de tinta individuales se mezclan entre sí, la forma del carácter resultante es de superior calidad que su equivalente matricial. La SQ-2000 produce sus caracteres a partir de una cuadrícula rectangular. En la modalidad borrador, los caracteres se componen en una cuadrícula de 15 por 24 puntos; en modalidad NLQ, la cuadrícula mantiene su tamaño original pero incluye 29 por 24 puntos para mejorar la calidad de la imagen impresa.

La cabeza de impresión está compuesta por 24 chorros dispuestos en dos filas de 12. Las filas están ligeramente desplazadas de modo que cada chorro se superpone ligeramente con su vecino de la otra fila, y es esta superposición lo que proporciona una imagen de impresión más fuerte. Al igual que sus parientes matriciales, la SQ-2000 también se puede utilizar para producir imágenes por mapa de bits, permitiendo volcar directamente en ella visualizaciones de pantalla en alta resolución. La velocidad de impresión es de 176 caracteres por segundo (cps) en modalidad borrador y 105 cps en modalidad NLQ.

Además de los accesorios de hoja cortada y alimentación por tracción, hay otras numerosas opciones disponibles. Éstas incluyen un tampón de entrada de 32 K que permite que el ordenador anfitrión vuelque rápidamente los datos a imprimir y luego prosiga con otras tareas mientras los datos se sacan del buffer y se imprimen automáticamente.



A la caza del papel
La SQ-2000 viene con un alimentador de hojas individuales, pero Epson también ofrece como opciones una unidad de tracción y una bandeja alimentadora. La unidad tractora posee los conocidos dientes de engranaje que empujan el papel a través de la cabeza de impresión. La unidad de la bandeja alimentadora será de gran utilidad a las pequeñas empresas, ya que permite alimentar con hojas de papel individuales la máquina desde dos bandejas que se pueden seleccionar independientemente, lo que resulta ideal para cartas personalizadas o informes

Además de las fuentes disponibles en la máquina estándar, se pueden añadir otras ROM.

Esta impresora es una compra ideal para el usuario serio de micros, porque es capaz de llevar a cabo las principales tareas de impresión que se requieren en el entorno de una pequeña empresa. Es de agradecer, especialmente, la reducción del nivel de ruido. Dado que su precio la coloca un poco fuera del alcance del bolsillo de la mayoría de los usuarios de ordenadores personales, la esperada introducción de una versión de precio reducido se hace sumamente atractiva.

EPSON SQ-2000

DIMENSIONES
620 × 297 × 188 mm

INTERFACES
En paralelo Centronics, RS232C, IEEE 488

VELOCIDAD
176 cps en modalidad borrador, 105 cps en modalidad NLQ *(near letter quality*: calidad casi de impresión)

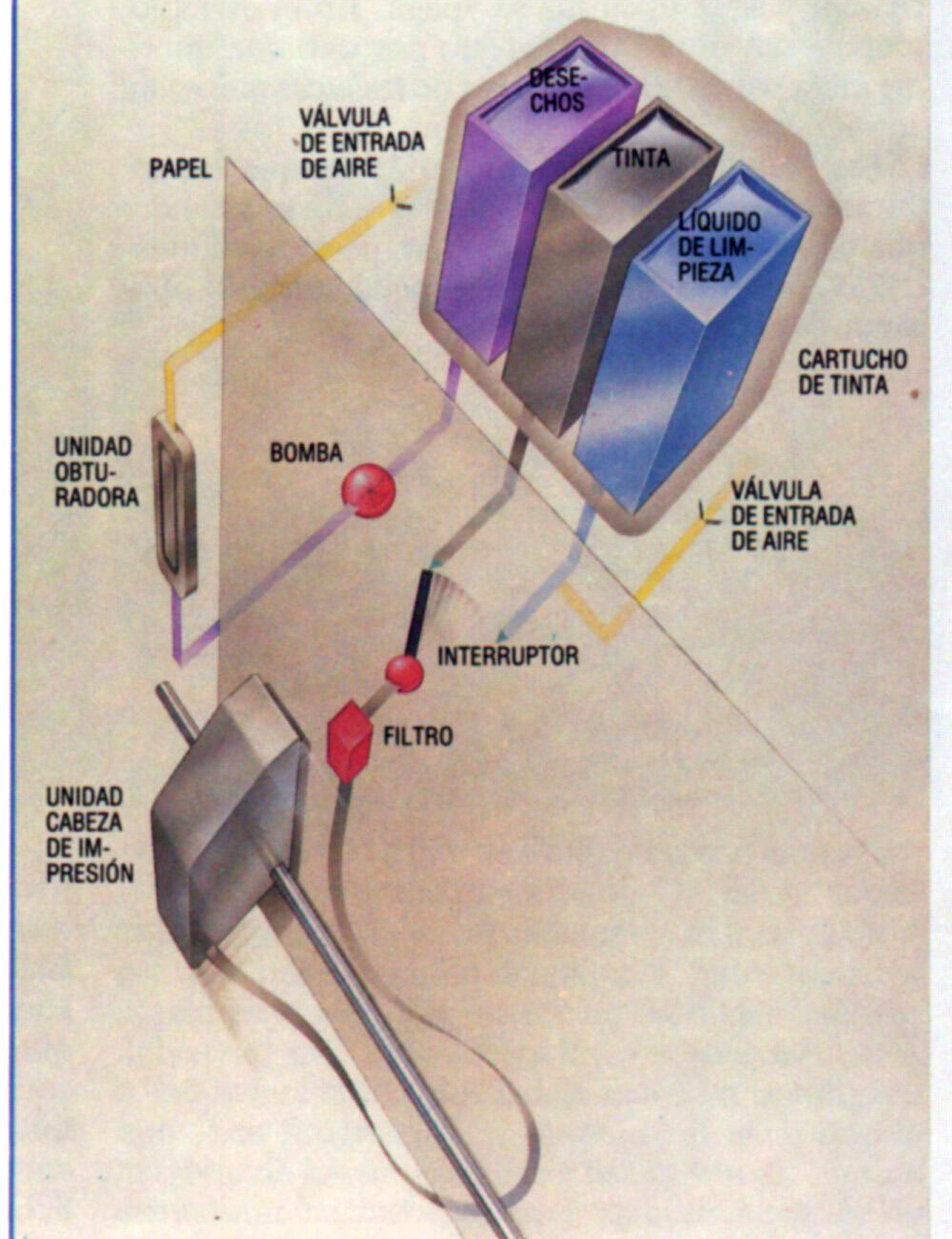
ALIMENTACION DE PAPEL
Por fricción o por tracción

ANCHURA DEL CARRO
140 mm mínimo, 406 mm máximo

DOCUMENTACION
Manual conciso y bien escrito, con elaborados ejemplos de generación de caracteres definidos por el usuario y gráficos por mapa de bits, apéndices completos de códigos de control y juego de caracteres

VENTAJAS
Combina la flexibilidad y la velocidad de las impresoras matriciales con la calidad de impresión de las impresoras margarita, y es mucho más silenciosa que ambas. Puede realizar las principales tareas de impresión que se requieran en una pequeña empresa

DESVENTAJAS
Para evitar que la impresión salga emborronada hay que escoger cuidadosamente el tipo de papel. Su precio la hace poco accesible a la mayor parte de usuarios de ordenadores personales



Haciendo la limpieza
El nuevo sistema de suministro de tinta desarrollado por Epson posee tres depósitos para tinta, líquido de limpieza y productos de desecho generados por el proceso de autolimpieza. El bombeo del líquido de limpieza a través del sistema mantiene a los tubos libres de posibles bloqueos, y una pequeña banda de goma situada a la izquierda del rodillo portapapel va frotando las boquillas del chorro de tinta por delante de la cabeza de impresión

Cortesía de Epson Ltd.

Epílogo

Finalmente demostraremos cómo la ampliabilidad de este lenguaje puede ser de gran utilidad en el desarrollo de un programa para controlar un sistema de semáforos

Como última palabra a nuestra serie dedicada al FORTH presentamos un programa que ilustra cabalmente la facilidad más útil de este lenguaje informático: su ampliabilidad. Como ya hemos visto a lo largo de la serie, este aspecto del FORTH, que da origen a estructuras fáciles de manipular y definir, permite su utilización para aplicaciones que otros lenguajes en realidad no pueden gestionar. Una de tales aplicaciones es el control de maquinarias, con el cual está relacionado el programa que proporcionamos a continuación.

Rojo, ámbar, verde

La máquina que estamos controlando en esta ocasión corresponde a un sistema de semáforos. Si estuviéramos haciéndolo en un plano real, lo haríamos a través de una puerta de salida en paralelo desde el ordenador, controlada mediante alguna palabra dependiente del sistema, como OUT (byte de salida --). Supongamos que las tres luces se controlan mediante tres bits de este byte de salida: el bit 0 para la roja, el bit 1 para la ámbar y el bit 2 para la verde. Si un bit es 1, entonces su luz se enciende, y si es 0, su luz se apaga. En el ejemplo, reemplazamos el OUT correcto por uno que imprime en la pantalla los nombres de las luces que están encendidas.

También necesitamos una palabra que espere durante un período dado. No hay ningún estándar que pueda cumplir este cometido, pero experimentando con su ordenador usted puede hacerlo utilizando bucles DO vacíos.

```
1 CONSTANT ROJO
2 CONSTANT AMBAR
4 CONSTANT VERDE

:OUT(codigo luces --)
DUP ROJO AND IF
  ."rojo"
ELSE
  ." "
THEN
DUP AMBAR AND IF
  ."ambar"
ELSE
  ." "
THEN
VERDE AND IF
  ."verde"
THEN
CR
;
VARIABLE LUCES (codigo ultimas luces
  escrito OUT)
  0 LUCES!
  0 OUT

:ENCENDIDA (codigo color --)
  LUCES @ OR
  DUP OUT
  LUCES!

;
APAGADA (codigo color --)
  NOT(usar-1 XOR en FORTH-79 o
    en ordenador ONE FORTH)
  LUCES @ AND
  DUP OUT
  LUCES!

;
VARIABLE BUCLESSEG
30000 BUCLESSEGS!(pruebe
  alterandolo)
:1SEG (espera un segundo)
  BUCLESSEGS @ 0 DO
  LOOP
;

:SEGUNDOS(segundos a esperar --)
  1 MAX 30 MIN (limite entre 1
    y 30 segundos)
  0 DO
    1SEG
  LOOP
;
;RUN
  ROJO APAGADO AMBAR APAGADO
  VERDE APAGADO
  5 0 DO
    ROJO ENCENDIDO 10 SEGUNDOS
    AMBAR ENCENDIDO 2 SEGUNDOS
    ROJO APAGADO AMBAR APAGADO
      VERDE ENCENDIDO
        10 SEGUNDOS
    VERDE APAGADO AMBAR ENCENDIDO
        2 SEGUNDOS
      AMBAR APAGADO
    LOOP
;
```

Está claro que ahora usted cuenta con un versátil lenguaje para controlar un sistema de semáforos, que podría ampliar fácilmente para cubrir un cruce completo. Es la ampliabilidad del FORTH lo que permite mezclar de una forma tan legible las partes de control de semáforos (como ROJO ENCENDIDO 10 SEGUNDOS) con las estructuras del programa.



Procesador de números

En el primero de estos tres artículos que dedicaremos al FORTRAN comprobaremos su gran capacidad aritmética

El FORTRAN se suele considerar como el primero de los lenguajes de alto nivel del tipo con el que estamos familiarizados en la actualidad. Se diseñó en un momento en que casi todos los ordenadores se utilizaban para tareas puramente numéricas, de modo que sus facilidades para manipular, por ejemplo, series de caracteres, son muy limitadas; pero así y todo sigue siendo uno de los pocos lenguajes capaces de manipular directamente números complejos. Su naturaleza matemática significa que se pueden escribir rutinas numéricas rápidas y eficaces, lo que lo convierte en un buen lenguaje para gráficos; asimismo, existen numerosas máquinas cuyos sistemas operativos se han escrito, al menos parcialmente, en FORTRAN.

Con el correr de los años se han ido construyendo muchas bibliotecas de subrutinas para una variedad de aplicaciones numéricas y éste es uno de los motivos de la continuada popularidad de lo que, al fin y al cabo, es un lenguaje muy anticuado, puesto que la forma más simple de utilizar este gran volumen de software es seguir escribiendo en FORTRAN. Parecería, entonces, que este lenguaje continuará siendo la principal herramienta de programación de la ingeniería y los laboratorios científicos.

Existen algunas versiones disponibles de FORTRAN para microordenadores, si bien pocas de ellas son implementaciones del estándar actual, el FORTRAN 77. La mayoría son versiones del estándar anterior, el FORTRAN IV, con ampliaciones moderadamente coherentes. Nos centraremos primero en el FORTRAN IV, porque ofrece la mejor idea de lo que es el lenguaje, y luego indicaremos las extensiones y matizaciones comunes al estándar, permitidas por la mayoría de las implementaciones para micros. El trazado de un programa en FORTRAN está determinado por el hecho de que la mayor parte de las entradas/salidas solían ser tarjetas perforadas de 80 columnas. Las ocho últimas columnas de tales tarjetas por lo general estaban reservadas para un número de identificación, lo que dejaba 72 columnas disponibles para cada sentencia de programa. En consecuencia, el FORTRAN sólo acepta una sentencia por línea y una línea por sentencia, con una longitud máxima de 72 caracteres. Si es necesario escribir una sentencia en más de una línea, entonces las líneas siguientes han de señalarse especialmente como líneas de continuación. Las 72 columnas restantes quedan aún más limitadas por el hecho de que las cinco primeras están reservadas para números de sentencia, la sexta columna se utiliza para indicar si la línea es o no una continuación, y la verdadera sentencia comienza en la columna siete.

Las líneas no se han de numerar como en un programa en BASIC, sino que a cualquier línea a la que se haga referencia en otra sentencia se le debe otorgar un número a modo de etiqueta. Estos números no deben seguir ninguna secuencia dada, sino que deben ser exclusivos. Una línea se puede utilizar como comentario colocándole una C en la columna uno. Los programas en FORTRAN, al igual que la mayoría de los programas en BASIC, son difíciles de estructurar correctamente, y carecen de tipos de datos y estructuras de control, lo que los hace dífici-

Evolución del lenguaje

Cronología de las versiones FORTRAN:

1954	Se formulan las primeras especificaciones del FORTRAN
1957	Primera publicación general de compiladores para FORTRAN (I) Formulación de especificaciones para el FORTRAN II
1958	Aparecen los primeros compiladores para FORTRAN II
1962	Se formulan las especificaciones para el FORTRAN IV
1963	Aparecen los compiladores para el FORTRAN IV
1977	Especificaciones para el nuevo FORTRAN 77
1978	Aparecen los primeros compiladores para el FORTRAN 77

Sperry Ltd.



Grace Hopper
Grace Hopper fue una de las fuerzas impulsoras del desarrollo de los primeros lenguajes de programación. A mediados de los años cincuenta, cuando trabajaba en Remington Rand, ayudó a producir uno de los primeros compiladores (A-2) y los subsiguientes lenguajes ARITHMATIC y FLOW-MATIC. Luego IBM refinó el concepto de lenguajes compilados en su versión final de FORTRAN, mientras que el FLOW-MATIC se convertiría posteriormente en el COBOL

les de leer y, por tanto, requieren la inclusión de comentarios como una importante característica.

Como en BASIC, pero no como en PASCAL, las variables no han de declararse; se las puede introducir en cualquier punto de un programa tan sólo con seleccionar un nuevo nombre. Sin embargo, si usted hace esto, entonces acepta la tipología de datos por defecto, que es que todos los nombres de variable empiecen con I, J, K, L, M o N si son de enteros, tomándose como reales todos los demás.

Normalmente, una de las principales dificultades para los programadores de FORTRAN recién iniciados son las complejidades de la manipulación de entrada/salida. El problema deriva de la dependencia básica de las tarjetas perforadas, con su formato estricto. Cada operación de E/S requiere dos sentencias: un READ o WRITE, que especifica tanto el dispositivo de E/S como las variables que se estén utilizando, y una sentencia FORMAT, que especifica los tipos y trazado exactos de los datos de la tarjeta, registro o línea de entrada de terminal. Por ej.:

```
    READ(1,100)X,I,ICHAR
100 FORMAT(F7.2,I4,A1)
```

El READ es la sentencia ejecutable que hará que se lleve a cabo una operación de entrada. El primer número dentro del paréntesis especifica el dispositivo de entrada que se está utilizando. La asignación de números a dispositivos depende de la implementación exacta y del hardware, pero un número habrá de referirse al terminal, otros a la impresora (sólo para salida), lector o perforador de tarjetas, o se le podría asignar a archivos en disco o cinta.

El segundo número dentro del paréntesis es un número de sentencia que identifica la sentencia FORMAT correspondiente. (Observe que, en el ejemplo, a la sentencia FORMAT se le ha dado el mismo número.) la sentencia FORMAT no es ejecutable; simplemente da información, de modo que se puede posicionar en cualquier lugar del programa. El número es la única forma de mostrar la relación, y es bastante permisible que más de un READ o WRITE se refieran al mismo FORMAT. Tras el paréntesis que sigue al READ se halla la lista de variables a las cuales se asignan los valores de entrada.

El FORMAT incluye una especificación para cada variable. Las principales especificaciones son:

- F: F7.2, por ejemplo, indica que el valor a entrar en X es real, ocupando las siete primeras columnas de la línea, registro o tarjeta con dos dígitos después del punto.
- I: I4 indica que el valor a entrar en I es un entero, ocupando las cuatro columnas siguientes de la línea.
- A: A1 indica que el valor de la siguiente columna es un carácter.

La única forma de que el FORTRAN pueda manipular caracteres es almacenándolos como variables de enteros. La cantidad máxima por variable depende del tamaño que tenga asignado el sistema para una variable de enteros, de modo que la utilización de enteros de 16 bits significa que sólo se permiten A1 y A2. No obstante, algunas versiones permiten almacenar caracteres también en variables de reales. Dado que las variables se pueden tratar tanto como numéricas como alfabéticas, y dado que la forma en que se almacenan los caracteres no siempre corresponde al código ASCII estricto, esto puede crear muchísima confusión. Las series de caracteres más largas, por ejemplo, se han de manipular utilizando matrices de enteros. En consecuencia, ¡el FORTRAN no es un lenguaje recomendable para aplicaciones que supongan tratamiento de textos!

Los verdaderos valores que se entran deben acomodarse a estas especificaciones, porque de lo contrario se generará un error en tiempo de ejecución. La mayoría de los sistemas modernos, sin embargo, permiten una entrada de la denominada *forma libre*, como READ (1,*)A,B,C o simplemente READ A,B,C, que está pensada específicamente para entradas por teclado, donde es difícil conseguir el espaciado exacto entre valores. Estas entradas de forma libre funcionan, entonces, de forma similar a la sentencia INPUT del BASIC. La salida se manipula de modo similar utilizando una sentencia WRITE:

```
    WRITE(1,200)A,B,C
200 FORMAT(3F7.2)
```

Observe cómo se pueden manipular las especificaciones repetidas sin tener que escribir una para cada variable.

Las series de texto se pueden hacer salir usando

Lógica comparativa

Operadores lógicos y conectores:

FORTRAN	BASIC
.EQ.	=
.NE.	<>
.GT.	>
.GE.	>=
.LT.	<
.LE.	<=
.NOT.	NOT
.AND.	AND
.OR.	OR

Eficiente con los números

FORTRAN IV admite varios tipos de datos numéricos:

- INTEGER (enteros) y REAL (reales), que poseen sus significados habituales
- DOUBLE PRECISION (doble precisión), que son números de punto flotante que utilizan el doble de la cantidad de palabras de memoria que los reales comunes
- COMPLEX (complejos) para números complejos con partes real e imaginaria
- LOGICAL (lógicos) equivalentes a un booleano del PASCAL, que puede tomar los valores .TRUE. (verdadero) o .FALSE. (falso) (hay que encerrarlos entre puntos)
- Los nombres de variables pueden tener hasta 6 caracteres de longitud, utilizando sólo caracteres alfabéticos o numéricos, debiendo ser alfabético el primero de ellos
- Las variables sólo se pueden declarar al principio del programa mediante una sentencia de declaración, como:

```
INTEGER NUM1, NUM2
LOGICAL FNSH
```



una especificación H especial, pero ésta es difícil de usar porque se debe enunciar la cantidad exacta de caracteres:

```
    WRITE(1,300)RESP
300 FORMAT(13HLA RESPUESTA ES,F7.2)
```

Nuevamente, muchos sistemas modernos admiten series entre comillas dentro del FORMAT:

```
    WRITE(1,300)RESP
300 FORMAT('LA RESPUESTA ES',F7.2)
```

pero esto no es FORTRAN estándar.

Cuando la salida se dirige a una impresora de líneas, el FORTRAN utiliza el primer carácter impreso para el control de papel, si bien los detalles exactos pueden variar de una instalación a otra. Un sistema típico, por ejemplo, podría utilizar un carácter de espacio para la espaciación simple normal, y los caracteres 1, 2, 3, etc., para dejar el adecuado número de líneas en blanco. Es un error común olvidar esto y perder el primer carácter de su salida mientras la impresora hace algo impredecible con el papel. Una salida hacia la impresora podría ser:

```
    WRITE(2,400)I,J,K,L,M,N
400 FORMAT(1H,2(I6,5X))
```

Observe el 1H extra, que controla el papel. La especificación X simplemente deja la cantidad de espacios en blanco especificada. Se puede encerrar entre paréntesis una secuencia de especificaciones con un multiplicador, como en 2(I6,5X).

En los ejemplos utilizaremos siempre un número 1 de dispositivo para indicar entrada/salida a través de un terminal, y el verdadero valor a utilizar aquí dependerá de la implementación empleada.

El FORTRAN es muy similar al BASIC en la forma en que establece la aritmética y la asignación, y la única diferencia importante es que el FORTRAN utiliza un asterisco doble (**) para la elevación a potencia en lugar de la lfecha hacia arriba (↑). En una expresión se pueden mezclar valores reales, enteros e incluso de doble posición, y asignarle el resultado a una variable de cualquier tipo. No obstante, se debe tener cuidado, ya que de lo contrario se podría encontrar con que todo el cálculo, o parte del mismo, se lleve a cabo en aritmética de enteros cuando se requiere un resultado de reales:

```
I=2
J=3
A=I/J
```

A tendrá el valor 0, porque la división se ha realizado en aritmética de enteros. Se proporciona una función FLOAT para convertir un valor entero en real, de modo que:

```
A=FLOAT(I)/FLOAT(J)
```

funcionará correctamente.

Habiendo sido diseñado como lenguaje aritmético, uno esperaría encontrar en el FORTRAN una gran variedad de funciones estándares. Hay demasiadas como para listarlas aquí, incluyendo todas las funciones trigonométricas y logarítmicas usuales, así como muchas otras que sólo tendrán algún significado para los matemáticos e ingenieros. Pero no sería arriesgado afirmar que el FORTRAN posee como función todas las operaciones numéricas estándares. Por último, la mayoría de las funciones se proporcionan en versiones tanto de reales como de doble precisión y, también, cuando es adecuado, en una versión de enteros.

Una de las principales críticas que se hacen al FORTRAN es su falta de estructuras de control, que en el FORTRAN 77 se ha paliado hasta cierto punto. La familiar sentencia GOTO se utiliza ampliamente y su destino puede ser cualquier sentencia numerada ejecutable (no una sentencia FORMAT).

Existen dos variables de sentencia IF: el IF lógico es similar al del BASIC, tomando la forma:

```
IF(expresion logica) sentencia ejecutable
```

(Los operadores lógicos y los conectores son una cuestión diferente.) La otra forma del IF es el de tipo aritmético, que suele considerarse el peor ejemplo de estructura de control. Éste asume la forma:

```
IF(expresion aritmetica)S1,S2,S3
```

donde S1, S2 y S3 son números de sentencia. El control se transfiere a la sentencia número S1 si el valor de la expresión aritmética es menor que cero, a S2 si es igual a cero y a S3 si es mayor que cero. Esto puede ser muy conveniente en tanto en cuanto permite una bifurcación en tres direcciones, pero puede dar lugar a un *código espagueti* aún peor que un GOTO normal.

Estudiante medio

```
C      PROGRAMA PARA LEER UN CONJUNTO
C      DE NUMEROS E IMPRIMIR SU
       MEDIA.
C      LA ENTRADA TERMINA
C      ENTRANDO UN NUMERO NEGATIVO
C
C      INICIALIZAR CUENTA Y SUMAR
C
       SUM=0
       ICOUNT=0
C
C      LEER SIGUIENTE NUMERO
C
100    READ(1,10)X
C
C      COMPROBAR SI ES NEGATIVO
C
       IF(X)300,200,200
C
C      NUMERO POSITIVO
C
200    ICOUNT=ICOUNT+1
       SUM=SUM+X
       GOTO 100
C
C        FIN DE LA ENTRADA
C
300    AVGE=SUM/FLOAT(ICOUNT)
       WRITE(1,20)AVGE
       STOP
C
C      FORMATS
C
10     FORMAT(F9.2)
20     FORMAT(10HEL PROMEDIO ES,F9.2)
       END
```

En pantalla

La visualización de caracteres es realizada por el OS del Amstrad de forma diferente de como visualiza los gráficos

La VDU para textos se encarga, entre otras cosas, de la lectura y la escritura de caracteres en pantalla, del cursor, de las ventanas y de los símbolos definibles por el usuario. El firmware del Amstrad permite el empleo en cualquier parte de la pantalla de hasta ocho corrientes con sus respectivas ventanas, cada una de las cuales con sus posibles tintas de fondo y letra *(paper/pen)*, su cursor, modo de fondo opaco o transparente, y la opción de escritura por caracteres de los gráficos. El tamaño de las ventanas de una corriente puede definirse con TXT_WIN_ENABLE, consultarse por medio de TXT_GET_WINDOW y borrarse con TXT_CLEAR_WINDOW.

Dos corrientes cualesquiera pueden tener, además, todos sus parámetros intercambiados gracias a TXT_SWAP_STREAMS. La mayoría de las rutinas del firmware actúan sobre la corriente actual; la entrada TXT_STR_SELECT está pensada para seleccionar la corriente actual, que quedará activa hasta que sea seleccionada otra corriente.

Los caracteres son visualizados en pantalla dentro de un cuadrado de 8×8 pixels. Existen varias entradas del firmware empleadas para imprimir un carácter, y una de las más importantes es la que mostramos en la tabla *Direcciones útiles*.

Por lo general, los caracteres ocupan cuadrados adyacentes y se imprimen en la posición actual del cursor para texto. Pero existe un modo especial, que se selecciona con TXT_SET_GRAPHIC, cuya utilidad consiste en trazar los caracteres en la posición del cursor de gráficos, con lo que es posible superponer texto y hacerlo aparecer en cualquier sitio de la pantalla.

La rutina principal, TXT_OUTPUT, se limita a guardar todos los registros antes de llamar la *indirección* TXT_OUT_ACTION. La rutina que queda apuntada por la indirección se encarga de tratar los códigos de control antes de pasar los caracteres visualizables a TXT_WR_CHAR, o a GRA_WR_CHAR si se activa la opción de escritura caracteres-gráficos. Se puede parchear la indirección para proporcionar un medio alternativo a la interpretación de códigos de control o a la escritura de caracteres hacia la pantalla.

La VDU para texto tiene un cursor correspondiente que indica la posición donde será visualizado el carácter siguiente. Se dispone de rutinas para activar y desactivar el cursor de texto y colocar o eliminar otro cursor en cualquier posición para carácter. Hemos detallado en la tabla citada las rutinas más útiles que guardan relación con el posicionamiento del cursor.

El programa *tabulador* permite el uso de paradas de tabulación a intervalos iguales según la variable TAB. El tamaño de la ventana actual se obtiene primero con una llamada (CALL) a TXT_GET_WINDOW, la cual proporciona en el registro D la columna extrema derecha, pero se hace que apunte a una posición posterior al margen derecho de columnas. Seguidamente, mediante TXT_GET_CURSOR se obtiene la posición efectiva del cursor; el registro H proporciona la columna del cursor, que se compara sucesivamente con una posible parada de tabulación hasta que encuentra la siguien-

Direcciones útiles

Rutina	Dirección	Observaciones
TXT_WIN_ENABLE	BB66	Entrar con H izq.; D derecha; L sup.; E inferior
TXT_GET_WINDOW	BB69	Salida con registros como antes
TXT_CLEAR_WINDOW	BB6C	Limpia ventana texto de la corr. actual
TXT_SWAP_STREAMS	BBB7	B y C retienen los números de las dos corrientes
TXT_STR_SELECT	BBB4	La corriente a seleccionar está en A
TXT_OUTPUT	BB5A	A retiene el carácter
TXT_WR_CHAR	BB5D	A retiene el carácter
TXT_RD_CHAR	BB60	Lee el carácter en el cursor y lo lleva a A
TXT_GET_CURSOR	BB78	H da la columna, L la fila
GRA_SET_ORIGIN	BBC9	Entrar con las coord. DE-X y HL-Y
GRA_SET_WIDTH	BBCF	Entrar con las coord. DE izq. y HL der.
GRA_SET_HEIGHT	BBD2	Entrar con las coord. DE sup. y HL inf.
SCR_SET_BASE	BC08	A retiene byte *hi* de la dir. base
SCR_SET_MODE	BC0E	A retiene núm. de modo (0,1 o 2)

Nota: Para usar adecuadamente estas rutinas, necesitará consultar la guía *Amstrad firmware specification*, editada por Amsoft (SOFT 158), que contiene las condiciones completas de salida y entrada de éstas y las restantes rutinas de firmware

te. Una vez conocida esa posible parada siguiente de tabulación, se efectúa una comprobación del espacio disponible para el campo de caracteres que sigue. Si hay espacio, entonces la columna del cursor se pone de modo que corresponda a la siguiente parada de tabulación, y en caso contrario se obliga al cursor a desplazarse a la línea siguiente mediante la especificación de una columna exterior a la ventana actual. Si fuera necesario, se puede parchear la rutina dentro de TXT_OUTPUT para que puedan admitirse otras paradas de tabulación distintas de las ocho columnas habituales.

Cada uno de los códigos ASCII comprendidos entre 0 y 255 tienen asociada una matriz que define los bits que conforman el carácter visualizado correspondiente. La matriz se compone de ocho bytes, y cada byte forma una línea del carácter. El dibujo muestra cómo se almacena en memoria la matriz del carácter correspondiente a CHR$224.

Asignación de pixels

Número de bit	7	6	5	4	3	2	1	0
Modo 2	1	2	3	4	5	6	7	8
Modo 1	1	2	3	4	1	2	3	4
Modo 0	1	2	1	2	1	2	1	2

Diferencias de modo
Cada byte en la memoria de pantalla retiene datos para un número de pixels entre dos y ocho, según el modo de visualización. El dibujo muestra a qué pixel corresponden los bits de cada byte en los distintos modos

Matriz de un carácter

Número del byte en la tabla de la matriz / Número de bit	7	6	5	4	3	2	1	0	
0	□	■	■	■	■	■	□	□	Fila superior
1	■	■	■	■	■	■	■	■	
2	■	□	□	■	■	□	□	■	
3	■	■	■	■	■	■	■	■	
4	■	□	■	■	■	■	□	■	
5	■	■	□	□	□	□	■	■	
6	■	■	■	■	■	■	■	■	
7	□	■	■	■	■	■	■	□	Fila inferior

□ Bit apagado
■ Bit encendido

Todo un carácter
Los caracteres son almacenados en matrices de ocho bytes, donde cada byte representa una fila de pixels del carácter. Los bits encendidos indican que un pixel ha de ser impreso en el color actual de la "pluma", y los apagados el color del "papel" (el fondo). El gráfico muestra la matriz del carácter 224 ASCII

El firmware presenta cuatro rutinas para determinar y redefinir estas matrices. Todas las matrices de los caracteres ASCII entre el 0 y el 247 se sitúan, por omisión, en la ROM inferior y por ello no son inicialmente redefinibles; los caracteres que van del 248 al 255 se copian en la RAM, donde puedan ser redefinidos. La rutina TXT_SET_M_TABLE puede emplearse para redefinir cualquier número de carácter comprendido entre el especificado y el 255, permitiendo así un carácter enteramente nuevo listo para su uso cuando se desee.

Los caracteres que han sido ubicados como redefinibles pueden aceptar la manipulación directa de sus matrices por medio de un programa del usuario, o pueden establecerse con la rutina TXT_SET_MATRIX que necesita ocho bytes apuntados por el registro HL y los copia en la matriz correspondiente a un carácter dado.

Se dispone también de la rutina TXT_GET_MATRIX para obtener la dirección de una matriz de carácter, así como de la rutina TXT_GET_M_TABLE para obtener tanto el primer carácter de una tabla definible por el usuario como su dirección.

El programa que presentamos, *Rotación de caracteres*, se sirve de TXT_GET_MATRIX y de TXT_SET_MATRIX para hacer rotar los caracteres en una de las cuatro direcciones. Esto se logra redefiniendo el carácter 248 para que tenga la misma matriz que el carácter requerido y manipulando después la matriz en el carácter 248 para la rotación deseada, antes de imprimirlo con TXT_OUTPUT. Se pueden conseguir efectos textuales muy provechosos enlazando este programa al BASIC.

La VDU de gráficos

La VDU de gráficos tiene por misión dibujar trazos en pantalla a una resolución de pixel. Se emplea una sola ventana de gráficos en la que las tintas de fondo y de primer plano pueden ser definidas independientemente de las usadas para la VDU de texto.

Existen tres sistemas de coordenadas que pueden servir para describir una posición de pixel en la pantalla. La resolución de pantalla efectiva es de 640×200 puntos en el modo 2, de 320×200 puntos en el modo 1 y de 160×200 puntos en el modo 0; éstas son las denominadas *coordenadas de base*. Sin embargo, cada punto en la pantalla tiene una dirección única que se emplea como referencia independientemente del modo actual. Los diferentes modos simplemente determinan cuánta área es asignada alrededor de un pixel al establecer ese punto. Así, en el modo 2 se establece un pixel, en el modo 1 se establecen dos y en el modo 0 se establecen cuatro.

Aunque la pantalla física tiene sólo una resolución de 200 puntos en vertical, se trata como si fuese de 400 puntos por el firmware, por lo que la proporción de 2:1 es aproximadamente aceptable. Por ello cada punto vertical corresponde a medio pixel y siempre se establecen al mismo tiempo las dos mitades de un pixel. Las filas de pixel 10 y 11, por ejemplo, corresponden ambas a la misma línea de pantalla.

Por lo tanto, la pantalla siempre es tratada como si tuviera una resolución de 640×400 en cualquier modo.

El sistema de coordenadas descrito sirve para las posiciones absolutas en toda la pantalla. Sin embargo, es posible definir una ventana de gráficos que emplee un tercio del total de las coordenadas para referenciar un punto.

El tamaño de la ventana se establece mediante GRA_SET_ORIGIN, GRA_SET_WIDTH y GRA_SET_HEIGHT; y puede determinarse empleando las

correspondientes GRA_GET_ORIGIN, GRA_GET_WIDTH y GRA_GET_W_HEIGHT. Decidido el uso de una ventana, todas las coordenadas se denomina coordenadas del usuario. El cursor de gráficos es controlado de dos maneras: el posicionamiento *absoluto* o el *relativo*. Si empleamos el método absoluto, el cursor se desplaza a los puntos especificados con respecto a su posición inicial. Las entradas empleadas para el posicionamiento del cursor se indican en un cuadro aparte.

La VDU de gráficos permite el trazado de pixels de tres modos: individualizados, formando caracteres o formando líneas. Todas las rutinas de trazados tienen la opción de trazado de puntos absolutos o la de puntos relacionados con la posición actual del cursor de gráficos. El cursor de gráficos se para en el último punto trazado, salvo cuando se trazan caracteres donde la posición vertical (ordenada) no cambia y la horizontal (abscisa) se mueve hasta el inicio del carácter siguiente. El cuadro mencionado detalla las entradas que se emplean para este fin.

Por *paquete de pantalla* se entiende una sección del firmware que proporciona el acceso a la pantalla en su nivel inferior. Maneja el direccionamiento de la pantalla, la implementación de la paleta de colores, la codificación y la decodificación de la tinta, la determinación del modo y el desfile de la pantalla, entre otras cosas.

La pantalla es un bloque de 16 K de RAM que puede ser ubicado en uno cualquiera de los cuatro bloques cercanos de 16 K en la memoria. Sólo pueden utilizarse los bloques que comienzan en $4000 y $C000, dado que los otros dos bloques contienen áreas del firmware que podrían quedar machacadas. Se puede conectar la pantalla entre dos bloques útiles por medio de la entrada SCR_SET_BASE; esta técnica puede servir para preparar una pantalla mientras se visualiza otra y hacer como si la actualización de la pantalla fuera instantánea.

Cada byte de memoria empleado para la pantalla contiene información sobre ocho posiciones físicas en pantalla. Éstas pueden corresponder a dos, cuatro u ocho pixels según sea el modo seleccionado por medio de SCR_SET_MODE.

Las asignaciones de pixels se ilustran en un diagrama que muestra cómo se determina cada pixel en un byte a través de los bits de los diferentes modos. Aunque al principio pueda parecer algo difícil, el sistema está de hecho diseñado de modo que, haciendo rotar un byte, se obtengan los bits necesarios para el pixel siguiente a la izquierda o a la derecha. Así, para poner un pixel dentro de un byte, se puede generar una máscara para establecer el pixel extremo izquierda y rotarla después hasta colocarla en la posición adecuada.

El camino más rápido para escribir en pantalla es el acceso a la memoria de pantalla, preferible al empleo de las VDU de textos o de gráficos. El paquete de pantalla proporciona varias rutinas para facilitar esta tarea.

Finalmente nuestro programa *Trazado rápido de un punto* es un ejemplo de listado que emplea estas entradas para trazar un punto físico en la pantalla. Obsérvese que no se puede usar junto con la pantalla de gráficos si se ha establecido una ventana.

La rutina pregunta antes que nada la dirección del byte que contiene información de la posición de pantalla especificada, y posteriormente pide la máscara para poner todos los pixels en ese byte con la tinta especificada. La máscara se manipula ahora para restablecer la posición requerida con tinta cero y volverla a poner en una nueva tinta.

El comPLOT de los 128 K

El Amstrad 6128 presenta unas sutiles mejoras de la instrucción PLOT según se implementaba en el CPC 464. El firmware en ambos modelos proporciona una rutina para seleccionar uno de los cuatro modos diferentes de trazado. Éstos son:

Modo	Actuación
0	Normal – pone nueva tinta al pixel
1	XOR – pone (nueva tinta XOR la existente) al pixel
2	AND – pone (nueva tinta AND la existente) al pixel
3	OR – pone (nueva tinta OR la existente) al pixel

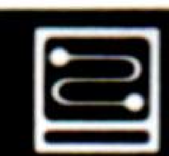
El modo XOR es muy útil, dado que permite trazar líneas y borrarlas sin perturbar el color de fondo si se desea. Por ej., si el color de fondo es el rojo (*paper* 3), el trazado de una línea en blanco (*pen* 4) producirá una línea en magenta (*pen* 7). Ésta puede ser borrada trazándola por segunda vez en *pen* 4, restaurando al mismo tiempo el color primitivo del fondo. Este modo de trazado es muy valioso en las rutinas de manejo de sprites, que necesitan desplazar un fondo sin perturbarlo. En el 6128, los modos de trazado se especifican añadiendo un parámetro más a la instrucción PLOT, pero esta facilidad no está prevista en el BASIC 1.0 del CPC 464. Todo lo que se necesita es una llamada a la rutina del firmware en &BC59, con el número del modo deseado en A

Tabulador

Este programa desplaza el cursor de texto respecto de la ventana activa hasta la columna siguiente, tabulada por medio de TAB. No hay condiciones de entrada y la rutina guarda todos los registros

```
tab:                    equ     15      ; columna de parada tabul.
get.window:             equ     #bb69   ; TXT_GET_WINDOW
get.cursor:             equ     #bb78   ; TXT_GET_CURSOR
get.column:             equ     #bb6f   ; TXT_SET_COLUMN
;
            push    af
            push    hl
            push    de
            call    get.window          ; entra tamaño ventana
            inc     d                   ; columna rt logica
            inc     d                   ; limite col. derecha
            call    get.cursor          ; toma pos. cursor
            ld      a, 1                ; inicio linea
;
loop:       add     tab                 ; toma col. tabul. siguiente
            cp      h
            jp      p, loop
;
            ld      h, a                ; lo guarda
            add     tab                 ; espacio para nueva col?
            cp      d                   ; comprueba límite

            jr      nc, setcol          ; no, obliga nueva linea
            ld      a, h                ; recoge nueva col.
setcol:
            call    set.column          ; establece nueva col.
            pop     de
            pop     hl
            pop     af
            ret
```

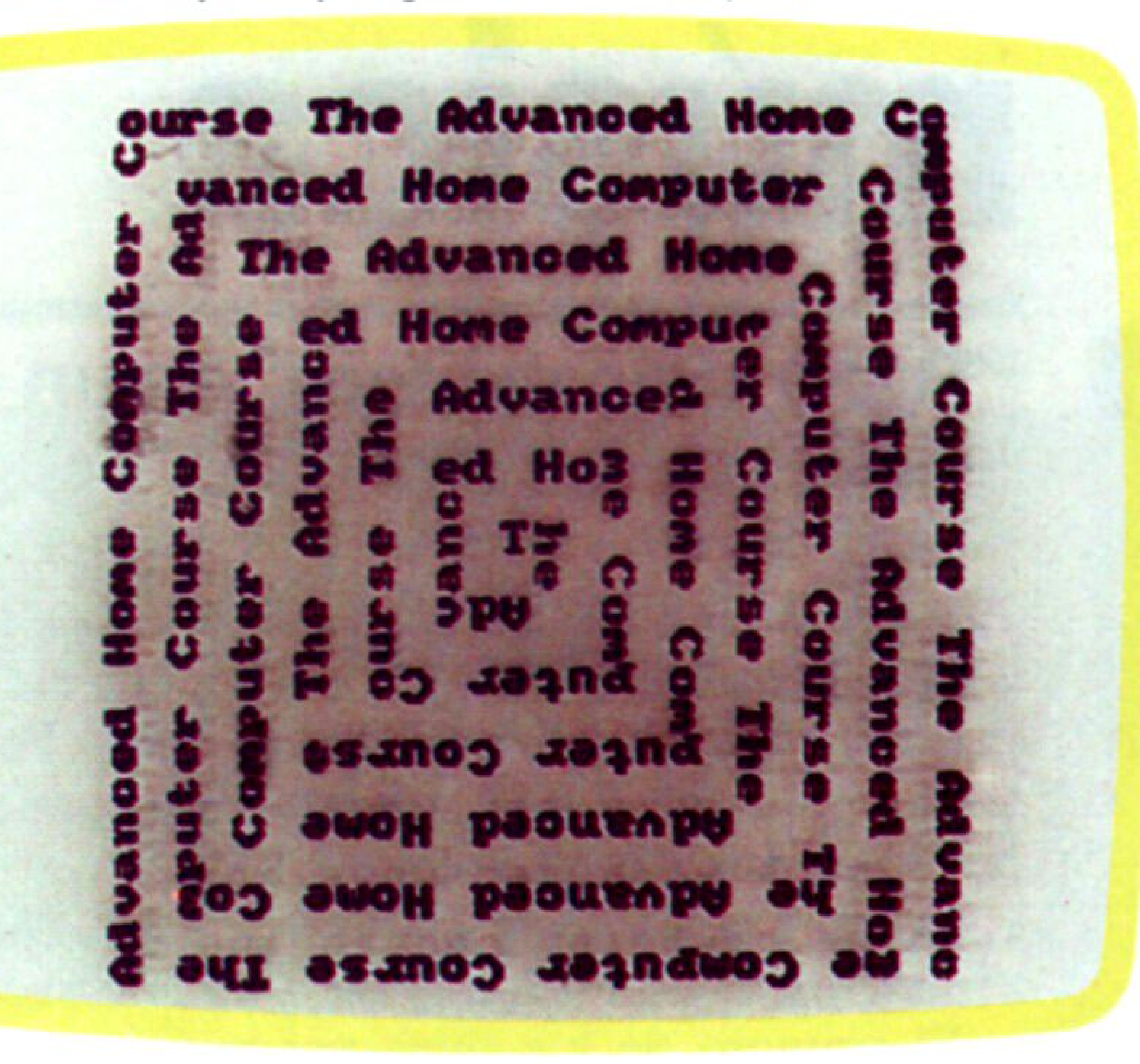


Rotación de caracteres

Este programa imprime un carácter tras rotar la matriz de caracteres hacia una de las cuatro esquinas. Primero se halla la dirección de la matriz de caracteres y después se emplea para redefinir el carácter 248, que normalmente es la primera entrada de la tabla de caracteres definidos por el usuario. Si este carácter no es redefinible, no ocurre nada

```
;
; Entry:                 IX+0 = Caracter por visualizar
;                        IX+2 = Esquina escogida 1 ... 4
                            1 = arriba
                            2 = atrás
                            3 = abajo
                            4 = normal
;
; Exist:
;                        Arrastre Verd – se imprime caracter
;                        Arrastre Falso = No se imprime carácter
;                        alterados AF, BC, DE, HL
;
txt.get.matrix:          equ     #bba5
txt.set.matrix:          equ     #bba8
txt.output:              equ     #bb5a
;
                         org     8000h
;
start:                   ld      a, (ix+0)          ; lee caracter
                         call    txt.get.matrix     ; toma dir. matriz
;
; Copia matriz en caracter 128 definible usuario
; después busca su dirección
;
           ld       a, 248                 ; caract. por redefinir
           call     txt.set.matrix         ; redefine matriz
           ret      nc                     ; nada si desactivado
;
           ld       a, 248                 ; caracter para rotar
           call     txt.get.matrix         ; halla matriz
;
; Rota la matriz caracter en sentido antihorario
; en mov. de 90 grados
;
           ld       (toprow), hl           ; guarda dirección
           ld       b, (ix+2)              ; lee esquina
;
lop3:      ld       d, 8                   ; bucle para 8 filas
;
lop2:      ld       e, 8                   ; bucle para 8 bits
;
lop1:      rlc      (hl)                   ; bit sig. en c
           rla                             ; guarda nueva fila
           inc      hl                     ; va a fila sig.
           dec      e
           jr       nz, lop1               ; busca fila sig.
;
           push     af                     ; nueva fila en pila
           dec      d
           ld       hl, (toprow)           ; busca bit sig.
           jr       nz, lop2
;
; Bucle para sacar de pila nueva matriz
;
           ld       e, 8
lop4:      pop      af                     ; toma fila sig.
           ld       (hl), a                ; va a matriz
           inc      h                      ; va a fila sig.
           dec      e
           jr       nz, lop4
;
           ld       hl, (toprow)
           djnz     lop3                   ; de nuevo, si es neces.
;
; Por ultimo visualiza caracter en pantalla
;
           ld       a, 248                 ; carac. por imprimir
           call     txt.output
           ret                             ; concluido
;
toprow:    defw     0                      ; inicio matriz
```

Texto en espiral
El programa de rotación de caracteres permite imprimir un texto verticalmente ¡y hasta de revés! La pantalla que mostramos da una idea de ello, aunque los efectos más brillantes se reservan para el usuario que emplee gráficos definidos por él



Trazado rápido de un punto

Esta rutina opera en todos los modos de pantalla y emplea el firmware para determinar una posición de memoria que contenga información del punto donde se ha de trazar el pixel

```
; Entry:       HL contiene la abscisa Y fisica
;              (0-199)
;              DE contiene la ordenada X fisica
;
;              (0-639)
;              A contiene la tinta asignada al punto
;
; Exit:        Se guardan todos los registros
;
dotpos:        equ       #bc1d          ; SCR_DOT_POSITION
encode:        equ       #bc2c          ; SCR_INK_ENCODE
;
plotter:
               push de            ; guarda los registros
               push hl
               push bc
               push af
;
               call dotpos                ; habla byte de memoria
               pop af                     ; repone la tinta en a
               push af
;
; HL contiene ahora la dirección del byte
; y C contiene la máscara necesaria para poner
; solo el pixel deseado
;
               call encode                ; halla la masc. tinta
;
; La mascara para poner todos los pixels con tinta nueva esta en a
;
               and       c              ; solo quiero este pixel
;
               ld        b, a           ; guarda numero tinta nueva
               ld        d, (hl)        ; toma posicion actual
               ld        a, c           ; toma mascara pixel
               camp      a              ; la invierte
               and       d              ; pone el pixel a tinta 0
;
               or        b              ; enmascara nuevo color
               ld        (hl), a        ; y lo almacena
;
               pop       af             ; restaura los registros
               pop       bc             ;
               pop       hl
               pop       de
;
               ret
;
```

Cree usted su propia música

"The music system" es un paquete de alta calidad que se diferencia nítidamente del software musical producido hasta ahora

Visualización notable
La visualización es apenas una de las muchas características atractivas de «The music system», valiéndose de iconos y menús diseñados con claridad para ayudar al músico. Esta instantánea está tomada de la opción «Keyboard»

The music system, de Island Software, posee un juego de cinco opciones, a cada una de las cuales se puede acceder fácilmente desde disco. Por el contrario, la carga desde cassette se complica por el hecho de que hay que manejar dos cintas. Pero con sus ventanas, iconos y gráficos de estilo Macintosh, el sistema es a la vez amable y muy atractivo.

El primero de los cinco programas, y el que se suele usar primero, es el *Editor*, que la empresa describe como el "procesador de notas de un músico". Éste hace que el proceso de escribir o modificar música resulte excepcionalmente sencillo. La pantalla visualiza pentagramas de bajo y soprano, en los que se escriben las notas; usted cambia el valor de las notas, mientras las líneas de los compases se insertan automáticamente según una clave de tiempo preseleccionada. Cuando ha compuesto una melodía, o cuando simplemente quiere escuchar unos pocos compases, pulsa la tecla Tab, que inicializará la reproducción de audio acompañada por una visualización que va desfilando por la pantalla.

Una de las principales cualidades de este sistema reside en que es a la vez adecuado para el recién iniciado y una gran ayuda para el músico ya experimentado. El principiante puede ver y oír exactamente lo que está sucediendo en cualquier etapa de la composición, y los errores se detectan fácilmente, haciendo que la corrección sea rápida y eficaz.

The song and sound library (Biblioteca de canciones y sonido) ofrece una gama de ritmos de apoyo para usar durante la composición, así como numerosas melodías muy conocidas que se pueden tocar y alterar, incluyendo la *Sonata para piano en Mi Bemol* de Mozart y *El vuelo del moscardón*. El *tempo* se puede regular desde un pausado paso de 30 compases por minuto hasta la vertiginosa velocidad de 200 compases por minuto, que ofrecería algunos resultados electrizantes si se aplicara a una fuga de Bach. Si esa velocidad le parece un tanto extravagante, siempre puede poner un ritmo suave después de la *Sonata* de Mozart.

La opción *Synthesiser* (Sintetizador), que se utiliza junto con las opciones *Editor* y *Keyboard* (Teclado), puede crear cualquier sonido que usted desee. Cada sonido se define estableciendo los parámetros de una envolvente, con un máximo de 30 envolventes. La versión en disco también ofrece una opción de 15 efectos de percusión distintos. Al igual que con los otros módulos, la edición es simple. Cada parámetro posee un propio icono en pantalla, y se pueden escuchar los sonidos a medida que se van entrando. Es fácil visualizar gráficos de frecuencia y amplitud mientras los sonidos de *Synthesiser* se pueden guardar en dos archivos y después cargar en el *Editor* o el *Keyboard*.

La opción *Keyboard* visualiza las dos octavas centrales de un teclado de piano completo, actuando el grupo QWERTY del teclado del micro como las teclas blancas, y las teclas numéricas como las negras. Encima del teclado hay una ventana que contiene los iconos de envolvente y de volumen.

Una de las características más notables de este módulo es su simulación de una grabadora de cinta de cuatro pistas. Las pistas de percusión y apoyo se pueden utilizar como fondo y componer música sobre ellas, con lo que usted puede experimentar con su música con gran facilidad.

The music system constituye un programa complejo e incluye dos exhaustivos manuales que es necesario estudiar para sacar el máximo provecho del paquete. Sin embargo, una vez cargado y en plena ejecución, las ventanas e iconos demuestran instantáneamente la sencillez y la eficacia del sistema.

¡Musica maestro!
«The music system» le ofrece al músico con o sin experiencia un control total sobre las facilidades de sonido de los ordenadores BBC Modelo B o Commodore 64. El sistema viene en dos paquetes: el primero contiene los módulos del sintetizador y el teclado; el segundo incluye las facilidades de editor y de impresión

Marcus Wilson-Smith

The music system: Para el BBC Micro Modelo B y el Commodore 64. Cassette Pack 1: consta de Synthesiser, Keyboard, Song & Sound Library; Cassette Pack 2: consta de Editor, Printout, Song & Sound Library
Editado por: Island Logic Limited, 22 St Peter's Square, London W6 9NW, Gran Bretaña
Formato: Cassette
Palancas de mando: No se necesitan


mi computer
Editorial Delta, S.A.


10105
9 788485 822836